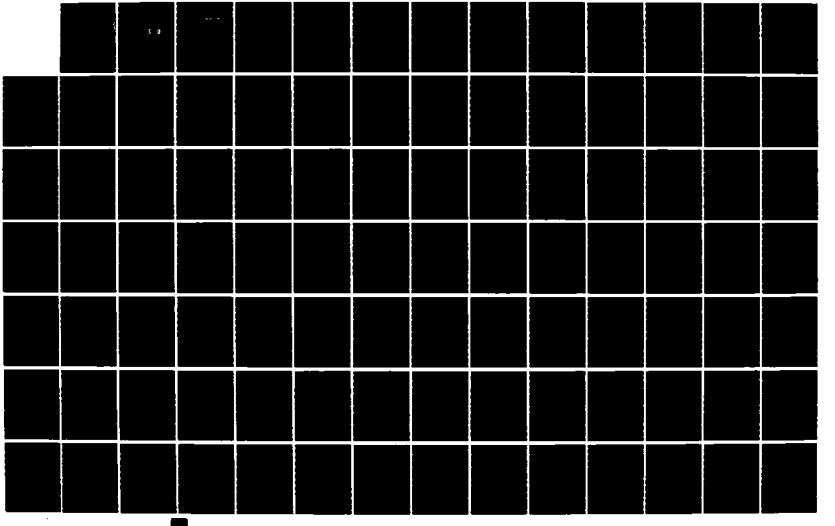
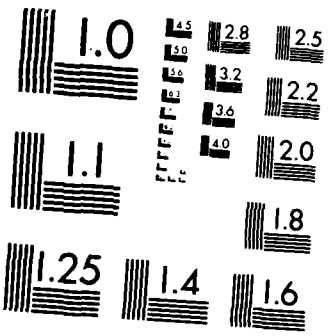
AD-A173 758 POLAR USER'S MANUAL(U) S-CUBED LA JOLLA CA 1/3
J R LILLEY ET AL. OCT 85 SSS-R-86-7563 AFGL-TR-85-0246
F19628-82-C-0081

UNCLASSIFIED F/G 22/1 NL





AFGL-TR-85-0246

Polar User's Manual

John R. Lilley, Jr
David L. Cooke
Gary A. Jongeward
Ira Katz

S-CUBED
P.O. Box 1620
La Jolla, CA 92038-1620

October 1985

Final Report
May 1982 - September 1985

AIR FORCE GEOPHYSICS LABORATORY
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
HANSCOM AIR FORCE BASE, MASSACHUSETTS 01731

This report has been reviewed by the ESD Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS).

This technical report has been reviewed and is approved for publication.

ALLEN RUBIN
Contract Manager

CHARLES P. PIKE
Branch Chief

FOR THE COMMANDER

RITA C. SAGALYN
Division Director

Qualified requestors may obtain additional copies from the Defense Technical Information Center. All others should apply to the National Technical Information Service.

If your address has changed, or if you wish to be removed from the mailing list, or if the addressee is no longer employed by your organization, please notify AFGL/DAA, Hanscom AFB, MA 01731. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD-A173 258

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) SSS-R-86-7563	5. MONITORING ORGANIZATION REPORT NUMBER(S) AFGL-TR-85-0246
6a. NAME OF PERFORMING ORGANIZATION S-CUBED 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Air Force Geophysics Laboratory
6c. ADDRESS (City, State, and ZIP Code) P.O. Box 1620 La Jolla, CA 92038-1620	7b. ADDRESS (City, State, and ZIP Code) Hanscom AFB, MA 01731
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Air Force Geophysics Laboratory 8b OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F19628-82-C-0081
8c. ADDRESS (City, State, and ZIP Code) Hanscom Air Force Base, MA 01731	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 62101F; PROJECT NO 7661; TASK NO 11; WORK UNIT ACCESSION NO. AA

11 TITLE (Include Security Classification)
POLAR USER'S MANUAL

12 PERSONAL AUTHOR(S)
Lilley, Jr., John R., Cooke, David L., Jongeward, Gary A., Katz, Ira

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final Report	FROM 5/1982 TO 9/1985	October 1985	282

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Spacecraft Charging; Auroral Ionosphere; 3-D; Flow; Finite Elements; Poisson Solution; POLAR Computer Code.

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

This report documents the physical principles and computational algorithms of the POLAR Code. POLAR models in three-dimensions the interactions of large spacecraft with the plasma environment in low polar orbit. It includes models of space charge limited particle collection, satellite wakes, the polar auroral environment, magnetic field effects, spacecraft surface charging, particle beam effects and sheath ionization.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL Dr. Allen G. Rubin	22b. TELEPHONE (Include Area Code) (617) 377-2933 22c OFFICE SYMBOL AFGL/PHK

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONCLUDED)

LIST OF ILLUSTRATIONS

TABLE OF CONTENTS (CONCLUDED)

LIST OF ILLUSTRATIONS (CONCLUDED)

LIST OF TABLES

1. INTRODUCTION

POLAR is a set of computer programs designed to predict the electrical interactions between the natural environment and a large spacecraft in polar earth orbit. POLAR consists of many complex physical models which have been converted to algorithms and connected by an executive structure. Since there are a wide variety of spacecraft and environments, POLAR has been written with maximum flexibility and applicability in mind. However, to allow for when a model may prove inadequate, POLAR has been designed with a high degree of modularity to enable changes in the physical models and algorithms to be made quickly and reliably. The documentation is also designed modularly so that code modifications can be documented immediately. Thus this manual is intended to be a living document, accurately reflecting the most current status of the POLAR code. This modularity does make for difficult reading, but we feel that the total information content is enhanced and that it is a valuable compromise. Since the models in POLAR are subject to change and replacement, it is important to have matched editions of the manual and the computer code. The edition date for this manual is July 1984, and it describes the edition of POLAR delivered to AFGL in July 1984.

1.10 CODE STRUCTURE

POLAR is written in FORTRAN in accordance with top down structural programming principles. It consists of four main programs and one attending utility program. Program one, called VEHICL, handles object definition. Program two, called ORIENT, is used to reorient an object and the grid. Program three, called NTERAK, actually calculates the spacecraft-plasma interaction and most of the physical models are contained in NTERAK. The fourth program is called SHONTL, and it controls plotting and information retrieval. The fifth program or utility, PLOTREAD, translates the machine independent output of SHONTL to local plot commands.

1.20 DOCUMENTATION

This document, as its code, is structured heirarchically. The description of POLAR goes from basic physics, to physical models, to algorithms, to coding structure, and finally, to operating constructions. As of this date (August 1983) many of the sections are missing as is much of the coding. As more coding comes on line it will be reflected in additions to this document.

3. PHYSICAL MODELS EMPLOYED IN THE POLAR CODE

The POLAR code makes various assumptions which enable it to perform three-dimensional charge calculations in relatively short Debye length plasmas. In this section we examine the component physical models and discuss their validity. While each is addressed separately, the code achieves a self-consistent solution by various levels of iteration. These are described more fully in Chapter 4, Computational Techniques. This chapter provides a so-called executive summary of the models as if the numerics were arbitrarily accurate. Numerical techniques are discussed in greater detail in Chapter 4.

One major, overriding assumption should be identified before the component by component description, which is that all time dependence on the scale of particle dynamics is ignored. This means that particles see spatially dependent but time independent fields for the period they are near the orbiting vehicle. As such, all plasma oscillations, including electron and ion modes are precluded. Thus, oscillations in the wake or at leading edges will not be predicted by the POLAR code.

3.10 THE POLAR PLASMA ENVIRONMENT

POLAR can model a wide variety of plasma environments from reasonable combinations of the following populations:

Ions;

Cool Maxwellian ions (input AMU).

Cool Maxwellian protons.

Both the protons and ions are assumed to be isotropic in the plasma frame. The relative densities are controlled by imputing the density ratio with the total constrained to equal the ambient electron density. Both populations have temperatures equal to the temperature 1 of the cool electrons.

Electrons:

Cool ambient Maxwellian, temperature 1, density 1.
Suprathermal, power law distribution of energies.
Hot Maxwellian, temperature 2, density 2.
Energetic, Gaussian distribution of energies.

The cool Maxwellian population is considered isotropic in the plasma frame. The other, more energetic populations may be given field-aligned and loss-cone distributions.

Limitation:

Currently, many of the options listed above are not consistently implemented throughout POLAR. Specifically, all electron populations are considered isotropic, and the proton population is ignored in the sheath model (3.60). These restrictions will be removed in future work.

3.20 PLASMA POTENTIALS

The other major assumption is that the only fields of major importance are the static electric fields obtainable from Poisson's equation and the earth's magnetic field. The only velocity related field included is that induced by $\underset{\sim}{V} \times \underset{\sim}{B}$ on conducting surfaces. The frame of reference is chosen to be the stationary plasma, so that $\underset{\sim}{V} \times \underset{\sim}{B}$ effects appear on the vehicle as boundary conditions. The plasma at infinity is defined to be at zero potential.

Plasma potentials are obtained from Poisson's equation

$$\nabla^2 \phi = \lambda^{-2}(n_i - n_e)$$

where λ is the Debye length ($\lambda^2 = \epsilon_0 \, kT/Ne^2$), and the charge density is that of the ambient plasma. Contributions from hot auroral electrons and particles backscattered from the vehicle are neglected.

3.30 PARTICLE DENSITIES

The electron density is assumed to be Maxwellian without any excluded orbits;

$$n_e = n_{e_o} e^{\phi/kT}$$

where n_{e_o} is the unperturbed cold component plasma density. The absence of excluded orbits implies neglecting potential barriers for electrons in the wake. The validity of this approximation has been studied using as an extreme case a disk moving infinitely fast with respect to thermal ions, but very slowly with respect to the electron thermal velocity. Such an object has rigorously no ion charge density in the wake and thus has the maximum negative space charge physically possible. Solving Poisson's equation for this case gives the maximum possible negative plasma potential. The central wake potential as a function of disk radius over Debye length is shown in Figure 3.30/1. We see that even for shuttle size objects that the peak space charge potential is less than 20 kT, or about 2 volts. If the surface boundary conditions are more negative than this wake space charge maximum, the potential in the wake will be monotonic and no electron orbits will be shadowed. This will clearly be the case for any case with substantial negative charging.

The ion density term is determined using one of two models, depending upon the local potential. At large distances from the object, where the potential is near plasma ground, that is substantially less than the ram energy of the ions, ion orbits are assumed to be unperturbed by electric fields. In this presheath region, ion densities are determined for both ions and protons, by the "neutral ion model" described next. A sheath edge is assumed to separate the presheath from a sheath region wherein electric fields dominate thermal effects. In the sheath region, ion densities are determined by sheath ion model (3.32, 3.60).

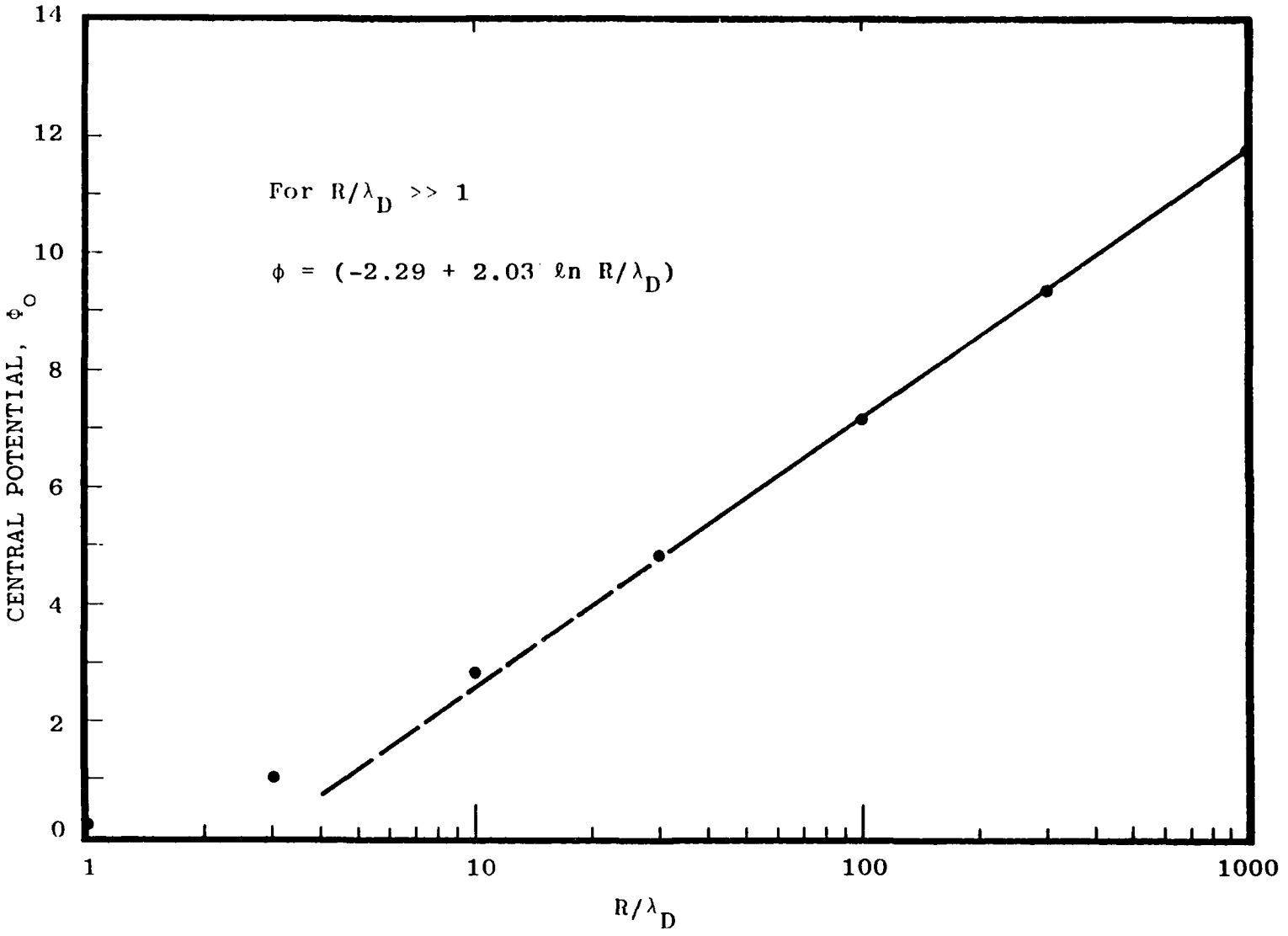


Figure 3.30/1. The central dimensionless potential (ϕ = eV/kT) of a cylindrical ion void of radius R, with equilibrium filling of ambient Maxwellian electrons.

It is important to note that the ion densities that result from the combined use of the neutral ion and sheath ion models are subject to certain limitations and shortcomings.

There is a low density or Laplace limit where the sheath edge is no longer sharply defined and electric fields extend far into the plasma. In this limit, the neutral ion approximation would fail, and thermal ion motion would be incorrectly ignored inside the sheath edge. This limit is characterized by a Debye screening length that is comparable to, or larger than, the object size. This does not mean that POLAR cannot provide useful results in the long Debye length limit, since the space charge coupling to the potentials is recuced by λ^{-2}. The user should, however, understand that ion densities and sheath ion currents (3.60) can be in error. There is also a short Debye length limitation that occurs when the Debye length is very much less than a zone size and object potentials are low. This combination can result in an object to sheath-edge distance that should be less than a zone, which is, of course, impossible to model accurately. This limitation is further discussed in Section 4.44, and illustrated in an example in Appendix C-3.

There is one further limitation of the ion density model that occurs when a sheath is significantly larger than an object. In this case, the ion densities outside the sheath will have been determined by the neutral ion model using the object as the only perturbation to ion orbits, where it should have used the sheath edge inasmuch as a sheath edge can be considered perfectly absorbing (Section 3.60). This problem is scheduled for resolution, but presently its effect should be minimal as far as object charging is concerned.

3.31 THE NEUTRAL ION APPROXIMATION

In some regions of space the electric fields may be weak, and potentials less than both the ion flow energy and the ion thermal energy. Here, ion motion may be approximated by that of a neutral particle where only collisions with absorbing objects perturb the ion density. The ion density in this region is calculated taking fully into account ion thermal velocities but ignoring bending of trajectories by electric or magnetic fields. This can be expressed by the following equation:

$$f_i(\underset{\sim}{x}, \underset{\sim}{v}) = g(\underset{\sim}{x}, \quad) \, f_{io}(\underset{\sim}{v})$$

where $f_i(\underset{\sim}{x}, \underset{\sim}{v})$ is the ion distribution function at a point $\underset{\sim}{x}$ in space for a velocity $\underset{\sim}{v}$ and $f_{io}(v)$ is the unperturbed velocity distribution function for a drifting Maxwellian (the assumed condition at an infinite distance from the vehicle). The function $g(\underset{\sim}{x}, \Omega)$ has value zero if a ray starting from x going in the direction Ω would strike the vehicle, it has value 1 otherwise. This function takes into account particles who cannot contribute to the local charge density because they run into the vehicle. The charge density is obtained by integration over velocities:

$$n_i(\underset{\sim}{x}) = \int f(\underset{\sim}{x},\underset{\sim}{v}) d\underset{\sim}{v} = \int_{\Omega} g(\underset{\sim}{x},\Omega) \left[\int_0^{\infty} f_{io}(v\,\Omega) v^2 dv \right] d\Omega$$

Since the potential varies logarithmically with density in the quasi-neutral region about large objects in short Debye length plasmas, a factor of 2 error in this density will lead to less than one-tenth volt error in the local potential; thus approximation is not expected to be a source of any large error.

3.32 SHEATH ION DENSITIES

Section 3.60 discusses the POLAR sheath model and the sharp edged sheath concept. The derivation of ion densities from trajectories within the sheath is explained in 4.42.5. What follows here is a brief outline.

Given a sheath edge, currents from "infinity" to the sheath edge are calculated analytically using orbit-limited theory. These currents are assigned to a set of "super particles" (Reference 3-5) that are tracked inwards from the sheath edge to the vehicle surface. Ion densities are determined from the product of the particle current and the time that a particle spends in an element, and by the focusing of trajectories.

3.40 INCIDENT CURRENTS

POLAR models a number of charged particle sources as responsible for surface and vehicle charging. These are ambient ions and electrons, energetic electrons, ion and electron generated secondary electrons, and photoelectrons.

POLAR is presently constructed to model plasma interactions with a negatively charged vehicle. Since the comments of Section 3.30 concerning potential barriers apply here as well, we can assume the ambient electrons to be repelled with no excluded orbits within the hemisphere of velocities impinging upon a surface. When these conditions are met, the velocity space integrals over a Maxwellian distribution decouple from the surface potential and we may write

$$j_e(V) = j_{eo}\, e^{eV/kT}$$

For the energetic electron sources, the current integrals are more involved. These are discussed further in Section 3.41 and presented in Sections 4.52.3 - 4.52.5.

The ions are presently the attracted specie in POLAR. The calculation of ion currents is discussed in Section 3.42.

3.41 INCIDENT ELECTRON CURRENTS

A statistical study (Ref. 3-1) of high latitude precipitating electrons has shown that these fluxes can be well represented by the following parametric expression

$$\Phi(E) = AE^{-\alpha} + Cn \frac{E}{(kT)^{3/2}} e^{-E/kT} + E\, Be^{-[(E-E_0)/\delta]^2} \qquad (3.41\text{-A})$$

These are the power law, hot Maxwellian and Gaussian distributions mentioned in Section 3.10, where $C = (2\, m_e)^{-1/2} \cdot \pi^{-3/2}$, and A, α, n, T, B, E_0 and δ are parameters determined by the particular shape of a spectrum. Here, $\Phi(E)$ has units of $\#/m^2 \cdot s \cdot str \cdot keV$. To apply these distributions to the charging of a surface, it is necessary to formulate the distribution function, f, at the surface. We start by disecting Eqs. (3.41-A). Equation (3.41-A) appears to assume a zero space potential because a factor of E (total energy) rather than K (kinetic energy) is used for the velocity or energy space differential volume unit. We next invoke the Vlasov equation to allow a mapping of f along a trajectory connecting the surface to "infinity". In doing so, we replace the one factor of E with K, while elsewhere setting $E = K + qV$ where q is the particle charge b, and V is the surface voltage. Thus, we write,

$$f(K,V,\psi) = A(\psi) \cdot (K + qV)^{-(\alpha+1)}$$

$$+ \sum_{e=1}^{2} \frac{F_e(\psi)}{(kT_e)^2 \pi} \cdot \exp(-(qV + K)/k\, T_e)$$

$$+ B(\psi) \exp(-(K - K_0)^2/\Delta^2)$$

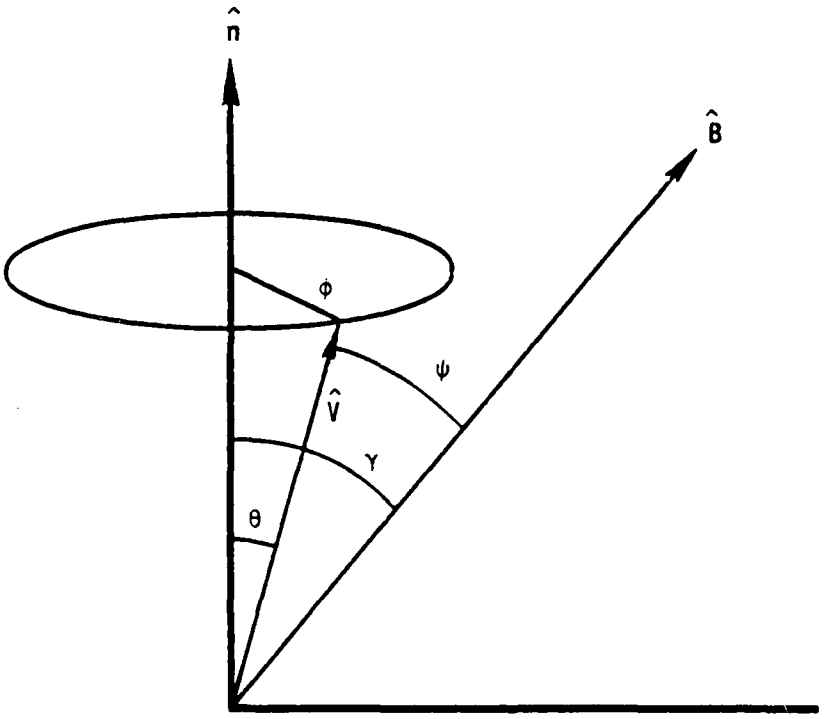
The index e covers both hot and cold electrons, and $F_e(\psi) = g(\psi)*n_e* \sqrt{kT/2\pi m}$ which is the thermal flux multiplied by a a function of the pitch angle. The net electron current density at a surface is, of course,

$$J = q \int_0^{\pi/2} d\phi \int_0^{2\pi} d\theta \int_L^U k\, dK\, f(K,V,\psi) \cos\theta \sin\theta \ .$$

The relation between the pitch angle ψ, and the spherical polar angles of the surface normal is:

$$\psi(\theta,\gamma,\phi) = \arccos[\cos\gamma \cos\phi + \sin\gamma \sin\theta \cos\phi]$$

where γ is the angle between the surface normal and the magnetic field. These angles are illustrated in Figure 3.41-1.



$$\psi(\theta, \gamma, \phi) = \cos^{-1} [\cos\gamma \cos\theta + \sin\gamma \sin\theta \cos\phi]$$

Figure 3.41-1.

It is important to note that in composing f at a surface from f at infinity, the angular factor in f, $A(\psi)$, $g(\psi)$, and $B(\psi)$, may evolve dramatically. Ultimately, POLAR will estimate the angular evolution of the electron distribution function, as this may be important for some narrow high energy distributions as well as necessary for the prediction of magnetic field effects. However, since the usual tendency is for the repelled specie distribution to broaden (POLAR currently assumes all surface and space potentials to be negative), our first guess will be to assume f to be isotropic at all surfaces.

The energy integration limits, U and L, are 0 and ∞ for the Gaussian and Maxwellian distributions, but a lower cutoff must be imposed on the power law distribution. This cutoff is physical in its origin as the electron currents are finite, but determining it accurately is not always possible. POLAR uses a 100 eV default cutoff which may be changed, or alternatively the total current for these electrons may be specified and a reasonable cutoff will be deduced by POLAR.

POLAR integrates each population separately. These integrations are described in Section 4.52.

3.42 ION SURFACE CURRENTS

POLAR presently considers two positively charged particle sources; one specie of ion (singly charged with a variable mass), and protons. As the attracted species, the calculation of the ion current density at a surface is a non-local problem which often depends critically upon the shape of the orbits that bring ions to the surface (Reference 3-2). That is, the problem is generally numerical with few exceptions that yield to analytic evaluation. POLAR calculates these currents as part of its sheath model, so the reader is referred to Section 3.60 for further information, and a brief discussion follows here.

External to a sheath edge "orbit-limited" (Reference 3-2 and Section 3.60) conditions may be assumed which will allow a flowing Maxwellian velocity distribution to be analytically integrated to find the ion currents to the sheath edge. These currents are assigned to representative particles that are traced inwards to the object surface to yield ion surface currents. This calculation is performed as a portion of the POLAR sheath model which is executed as the CURREN module of NTERAK (see Chapter 5 for POLAR code structure).

3.50 ELECTRICAL CHARGING

The electrical charging of the spacecraft is modeled using a circuit analogy. The plasma around the craft becomes a current source with a capacitance between the plasma and the object's surface. The spacecraft is modeled as a network of capacitors, resistors, and voltage sources. The basic charging equation is

$$\underset{\sim}{I}(t) = \underset{\approx}{C} \frac{d}{dt} \underset{\sim}{V}(t) - \underset{\approx}{\sigma}\, \underset{\sim}{V}(t) \qquad (3.50\text{-A})$$

where I is current, C capacitance, σ conductance, and V is voltage. Each surface (the smallest mesh unit sized square, triangular and rectangular building block) contributes a component to the current and voltage vectors.

Surface voltages are updated (integrated) by timestepping a finite difference approximation to Eq. (3.50-A) (4.51). This integration frequently proves to be difficult because of the wide range of capacitance that can occur in the $\underset{\approx}{C}$ matrix. For instance, a surface to plasma capacitance might be

$$\frac{C_\infty}{A} \simeq \frac{\epsilon_o}{R} \simeq 10 \text{ pf/m}^2$$

where A is the surface area and R is an effective radius, whereas the surface to conductor capacitance of a dielectric might be

$$\frac{C_b}{A} \simeq \frac{\epsilon}{d} \simeq 0.1 \text{ } \mu\text{f/m}^2 \text{ .}$$

Thus, a driving current of 10^{-5} A/m^2 would produce charging rates of 10^6 volts/sec and 10^2 volts/sec. Obviously, these two extremes would require different timesteps for a simple explicit integration.

The stability difference between the explicit and implicit forms can be demonstrated by a simple scalar analog.

Explicit: $C[V(t_2) - V(t_1)] = I(t_1) \cdot \Delta t$

Implicit: $C[V(t_2) - V(t_1)] = I(t_2) \cdot \Delta t$

Substituting

$$I(t_2) = I(t_1) + I'(V(t_2) - V(t_1)) \qquad (3.50\text{-B})$$

gives

$$V(t_2) - V(t_1) = \frac{I(t_1)\,\Delta t}{C - I'\Delta t}$$

If we take a case of $C = 10^{-11}$ f, $I(t_1) = 10^{-6}$ A, $\Delta t = 1$ sec, $I' = -10^{-8}$ A/volt we find

Explicit: $\Delta V = 10^5$ volts

Implicit: $\Delta V = 10^2$ volts

That the explicit answer is unstable is indicated by plugging ΔV into Eq. (3.50-B) giving $I(t_2) = -10^{-3}$ A (explicit) or $I(t_2) = 10^{-9}$ A (implicit).

POLAR utilizes both explicit and implicit timestepping to allow large timesteps for the large capacitances while maintaining accuracy and stability for the smaller capacitances. Details of implementation can be found in Sections 4.51 and 5.70.

REFERENCES, CHAPTER 3

3-1 Fontheim, E. G., K. Stasiewicz, M. O. Chandler, r.s.b. Ong, and E. Gombosi, "Statistical Study of Precipitating Electrons."

3.60 THE POLAR SHEATH MODEL

The concept of a plasma sheath requires definition. In general, the plasma sheath can be defined to be the region of non-neutral charge density that shields a charged body from distant plasma. A more precise definition should distinguish between an "orbit-limited" sheath and a "space charge-limited" sheath. Investigations into current collection by Langmuir probes (Ref. 3-2, 3-3) in the long Debye length limit, have shown that current collection is orbit-limited, i.e., on the surface of a probe, distribution functions are filled over a hemisphere, and are related to the distant plasma distribution function by constants of the motion or "orbits". As the Debye length is shortened, current collection remains orbit-limited until the charge density is sufficient to cause the electric potential to decrease faster than the inverse square of the radial distance. At this point, current collection is said to be space charge-limited.

An important feature of the orbit-limited sheath is that the particle currents to a surface are independent of the exact shape of the potential well, making it possible to derive general expressions for currents and densities. The opposite is true of the space charge-limited sheath where currents and densities must be calculated numerically by following trajectories in potential wells that must be consistent with the particle densities. The many approaches to this problem have been reviewed recently by Laframboise (Ref. 3-4).

POLAR must model the space charge-limited extreme, which dictates the use of some trajectory tracing. Efficiency is maintained by recognizing that orbit-limited conditions exist in the quasi-neutral region outside the sheath, and that trajectories must be followed only inside the sheath. POLAR thus makes a sharp sheath edge approximation to divide a problem into the two regimes. Fluxes from "infinity" to the sheath edge are calculated analytically using orbit-limited theory (Sections 3.42, 4.42.2). These fluxes are then assigned

(Section 4.42.3) to trajectories that are tracked (Section 4.42.4) through the sheath to determine particle densities in the sheath (Section 4.42.5) and surface currents (Section 4.53). Of course, for the self-consistent probe problem, POLAR must iterate between sheath density solutions and Poisson solutions, and for a charging problem a higher level of iteration updates the surface potentials and iterates with the sheath-Poisson solution.

The sheath edge is nominally chosen to be the 0.47 kT potential contour. For a spherical probe in a non-flowing plasma, this is consistent with previous investigations (Refs. 3-5, 3-6). In the presence of net plasma flow, POLAR maintains the 0.47 kV sheath edge choice, but it is presently not clear what the best choice for the sheath edge will be for high flow problems.

There are two major approximations made in the POLAR sheath model. The first is the so-called sharp sheath edge approximation. This assumes that there is a sharp boundary between non-neutral sheath and the surrounding quasi-neutral presheath region. For large objects in short Debye length plasmas this is a very good approximation. The other approximation is that thermal effects within the sheath are small, i.e., from a given position on the sheath boundary, that a single trajectory is adequate to represent all particles entering from that position. This implies that potentials exist in the sheath such that

$$\frac{\phi}{kT} >> 1$$

and that the electric fields near the sheath edge are sufficiently strong so that a velocity space element is accelerated rapidly and its thermal spread is small;

$$v_{th} \cdot \tau << L$$

where v_{th} is the ion thermal velocity, τ is the transit time from the sheath edge to the vehicle, of characteristic dimension L.

REFERENCES, CHAPTER 3

3-2 Laframboise, J. G., "Theory of Spherical and Cylindrical Langmuir Probes in a Collisionless, Maxwellian Plasma at Rest," UTIAS Report No. 100, 1966.

3-3 Laframboise, J. G. and L. W. Parker, "Probe Design for Orbit-Limited Current Collection," Phys. of Fluids, Vol. 16, N5, 1973.

3-4 Laframboise, J. G., "Is There a Good Way to Model Spacecraft Charging in the Presence of Space-Charge Coupling, Flow, and Magnetic Fields?," in Proceedings of the Air Force Geophysics Laboratory Workshop on Natural Charging of Large Space Structures in Near Earth Polar Orbit, AFGL-TR-83-0046, September 1982.

3-5 Parker, L. W., "Computations of Collisionless Flow Past a Charged Disk," NASA CR-144159, 1976.

3-6 Parrot, M.J.M., L.R.O. Storey, L. W. Parker and J. G. Laframboise," Theory of Cylindrical and Spherical Langmuir Probes in the Limit of Vanishing Debye Number," Phys. Fluids, 25(12), December 1982.

4. COMPUTATIONAL TECHNIQUES

This section describes the algorithms used to implement the physical model.

4.10 GRIDS - DISCRETIZATION OF SPACE

POLAR is a three-dimensional computer code; that is, its internal representation of space allows variations in all three coordinate directions. Since problem set-up can be extremely complex in three-dimensions, the choice was made to keep the spatial coordinate system as simple as possible. Space is divided uniformly into cubes. The computer code stores information about a large set of cubical volumes, called elements. It also stores values of electric potential for the corners of the elements. The corners are referred to as nodes.

4.11 STAGGERED MESH

The coordinate system used in POLAR is Cartesian. This greatly simplifies object definition and converting positive vectors into element locations. However, the types of problems POLAR is designed to handle are very anisotropic. The density and potential perturbations are along the wake direction and can extend for several vehicle diameters. To provide resolution in this wake region but not occupy an excessive amount of computer storage, a system of staggered meshes has been implemented. The staggered mesh consists of square layers of elements that are only one mesh unit deep in the z-direction. They are stacked upon each other so that the center of each layer is as close to the wake center as possible and still have the nodes at integer coordinates. Since this extension always is in the z-direction, the object coordinates can be transformed by a 90° rotation matrix so that an arbitrary Mach vector can be accommodated.

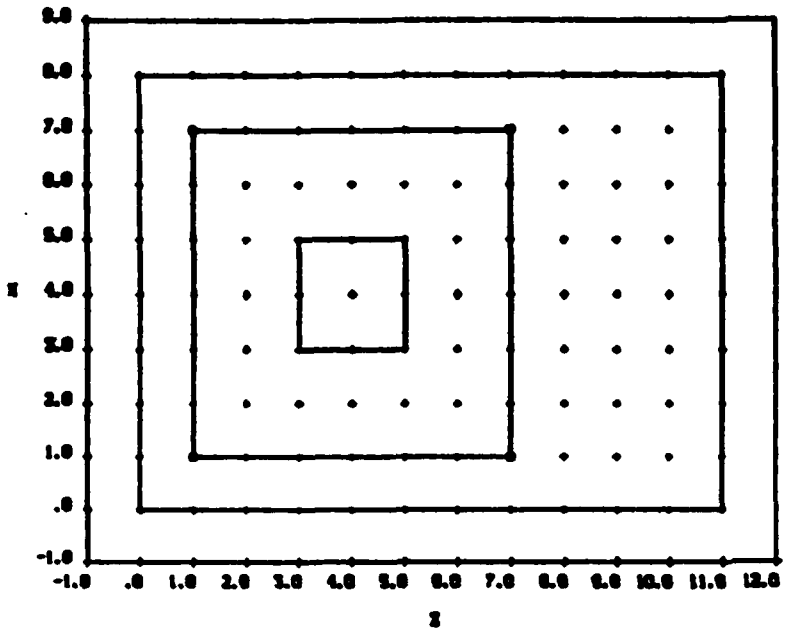
The problem space is unrestricted in the z-direction so that viable results may be obtained for large Mach vectors in lower density plasma.

Examples of grids for two objects are shown in Figure 4.11/1.

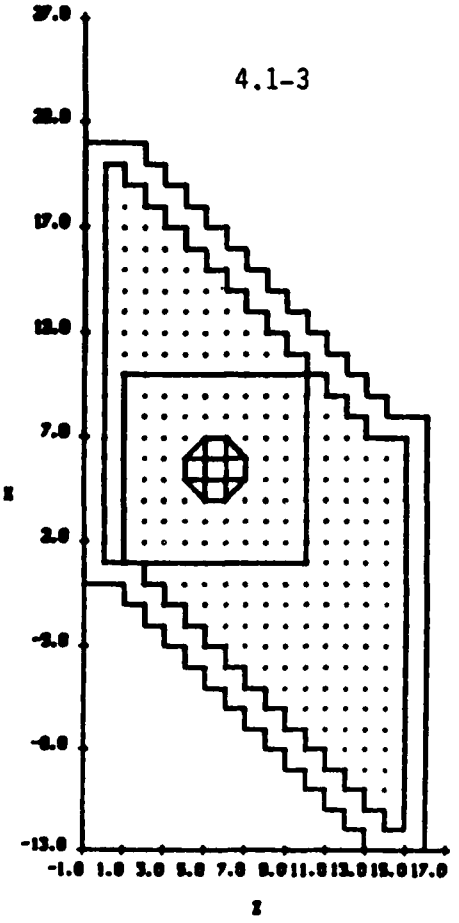
4.12. OBJECT GRID

The object grid is the space in which the object is defined. This subsection of the grid space is non-stepped and is a regular prism. The object must be defined so that it fits entirely within this space.

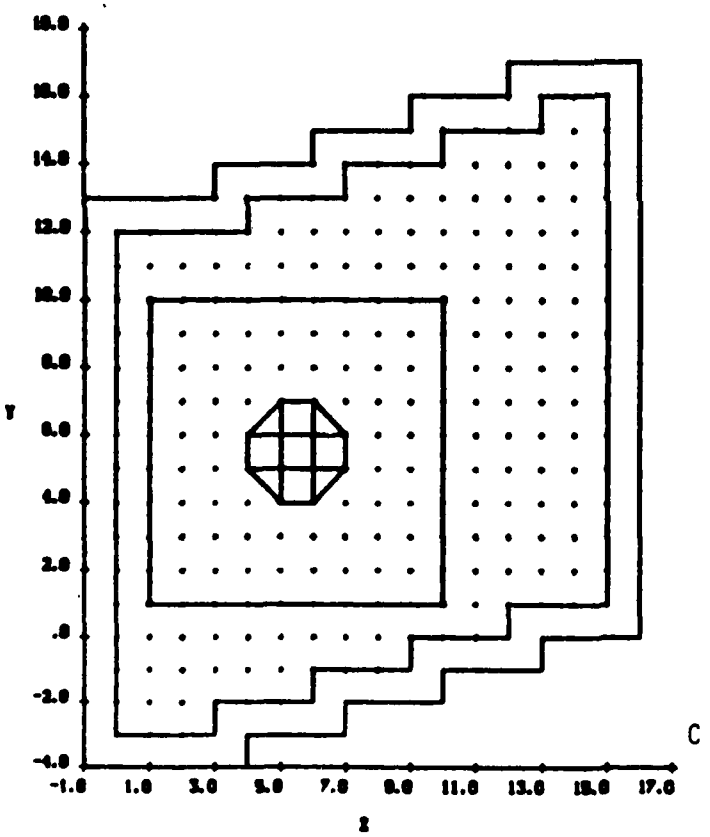
The object grid also serves as the coordinate reference point for the entire problem. The lowest point on the x, y and z axis (low leftmost corner) is defined to be the origin and has the coordinate value of (1,1,1).



A



B



C

Figures 4.11/1A. Cube moving to left; 4.11/1B. Rising quasi-sphere; 4.11/1C. Falling quasi-sphere.

4.21 FINITE ELEMENTS

4.21.1 GENERAL

Consider a charged object isolated in space. The potential ϕ everywhere is given by the solution to Poisson's equation

$$\nabla^2\phi = -\rho/\epsilon \tag{4.1}$$

The variational principle associated with this equation is given by:

$$\frac{\delta}{\delta\phi}\left[\left(\int dV\ \frac{1}{2}\ (\nabla\phi)^2 + \frac{\rho\phi}{\epsilon}\right) + \int_{C_S} \frac{\sigma\phi}{\epsilon}\ ds + \int_{C_B} \phi\ .\ \phi.dS'\right] = 0$$

where we integrate over both the object and boundary surfaces (C_S, C_B).

To simplify things for the purpose of illustration, let us fix the potentials on these surfaces and assume zero charge density. The equation then simplifies to

$$\frac{\delta}{\delta\phi}\int dV\ \frac{1}{2}\ (\nabla\phi)^2 = 0 \tag{4.2}$$

Equation (4.2) involves an integral over the volume of the computational space. One way to treat this integral is to divide the space up into finite cubic volume elements.

$$\int dV \, \frac{1}{2} (\nabla\phi)^2 = \sum_e \int_{V_e} dV_e \, \frac{1}{2} (\nabla\phi)^2$$

In this approach the potential ϕ is defined at each grid point, or node, defining the vertices of the elements. The potential inside each element is then trilinearly interpolated from the values of each of its eight vertices.

$$\phi^e (x,y,z) = \sum_{i \in e} N_i^{xyz} \, \phi_i$$

where "i" are the nodes of element "e"

$$\nabla\phi^e (x,y,z) = \sum_i \nabla N_i^{xyz} \, \phi_i$$

and

$$\int dV \, \frac{1}{2} (\nabla\phi)^2 = \sum_e \int dV_e \left| \sum_i \nabla N_i^e (x,y,z) \phi_i \right|^2$$

The quantity

$$W_{ij}^e = \int_e dV_e \, \nabla N_i^{xyz} \cdot \nabla N_j^{xyz} \qquad (4.3)$$

is completely defined just by knowledge of the shape of the element "e" (i.e., whether the cube is empty or partially filled). For screening purposes, we also require

$$V_{ij}^e = \int dV_e \, N_i^{xyz} \, N_j^{xyz} \qquad (4.4)$$

The variational principle therefore becomes:

$$\frac{\delta}{\delta\phi}\left[\sum_j \sum_i \sum_e W_{ij}^e \phi_i \phi_j = 0\right] \qquad (4.5)$$

Let $\sum_e W_{ij}^e = M_{ij}$. Equation (4.4) becomes

$$\frac{\delta}{\delta\phi}\left[\sum_{ij} \phi_i M_{ij} \phi_j\right] = 0 = \underset{\approx}{M}\ \underset{\sim}{\phi}$$

Thus the set of ϕ values at each node ($\underset{\sim}{\phi}$) that satisfy $\underset{\approx}{M}\ \underset{\sim}{\phi}$ is the solution to the Poisson equation (4.1) under conditions of fixed object and boundary potentials and zero charge density.

We may solve Eq. (4.4) iteratively. Our initial choice of $\underset{\sim}{\phi}$ will yield a residual $\underset{\sim}{r}$

$$\underset{\approx}{M}\ \underset{\sim}{\phi} = -\underset{\sim}{r}$$

The iterative scheme used is the Conjugate Gradient technique. It is based on the following equations:

$$\underset{\sim}{r}_o = -\underset{\approx}{M}\ \underset{\sim}{\phi}^o$$

$$\underset{\sim}{u} = \underset{\sim}{r}^o$$

$$a^i = (r^i, r^i)/(\underset{\sim}{u}^i, \underset{\approx}{M}\ \underset{\sim}{u}^i)$$

$$\underset{\sim}{\phi}^{i+1} = \underset{\sim}{\phi}^i + a^i\ \underset{\sim}{u}^i$$

$$\underset{\sim}{r}^{i+1} = \underset{\sim}{r}^i - a^i\ \underset{\approx}{M}\ \underset{\sim}{u}^i$$

$$b^i = (r^{i+1}, r^{i+1})/(r^i, r^i)$$

$$\underset{\sim}{u}^{i+1} = \underset{\sim}{r}^{i+1} + b^i \underset{\sim}{u}^i$$

These equations may be iterated upon until the resultant $\underset{\sim}{\phi}$ vector becomes the solution to Poisson's equation.

The major computational operation in the iterative set of equations is the evaluation of the matrix-vector product $\underset{\approx}{M}\,\underset{\sim}{u}$. The vectors $\underset{\sim}{\phi}$, $\underset{\sim}{u}$, and $\underset{\sim}{r}$ all have the same number of elements as the number of grid points. M contains the square of this number. Such a huge array is impractical to store all at once and so M u is evaluated using the following implicit algorithm

$$\underset{\sim}{r} = \sum_e \underset{\sim}{r}_e = \sum_e \underset{\approx}{w}_e \underset{\sim}{u}$$

The $\underset{\approx}{w}_e$ matrices are of reasonable dimension, for example, 8 x 8 for an empty cube. The residual $\underset{\sim}{r}$ is constructed element by element and then summed. These "weight" matrices $\underset{\approx}{w}_e$ may be calculated analytically for each type of empty or partially filled volume element, allowed by POLAR. There are seven of these. Filled cells are not included in the potential calculation. This is how POLAR treats filled, partially filled and empty elements, differently.

4.21.2 SPECIAL CELLS

The geometry of NASCAP/POLAR objects can be treated in terms of a relatively small number of volume cells. Each type has a maximum of eight corner nodes, plus a node at the center of each possible surface pointing into the element. The most common volume element (designated type 0) is the empty cube with no surfaces. Other elements are:

a. The empty cube with up to 6 surfaces (also type 0);
b. The wedge element (type 1);
c. The empty element with a diagonal line (produced by a right-triangle surface or a slanted thin plate) on one face (type 2);
d. The tetrahedron (type 3);
e. The truncated cube (type 4);
f. The slanted thin plate (type 5).

Each element is characterized by (1) a standard orientation; (2) a set of interpolation functions (see 4.21.21 for treatment of surface nodes); (3) the matrix $\underset{\approx}{W}$, given by Eq. 4.3, which represents the operator $-\nabla^2$ in Poisson's equation; and (4) the matrix $\underset{\approx}{V}$, defined by Eq. 4.4, which handles the screening part of Poisson's equation.

4.21.21 INTERPOLATION FUNCTIONS FOR FACE-CENTERED SURFACE NODES (FCSN'S)

In constructing the matrices $\underset{\sim}{\underset{\sim}{W}}$ and $\underset{\sim}{\underset{\sim}{V}}$ for volume cells with FCSN's it is convenient to work with the vector $\underset{\sim}{\psi} = \underset{\sim}{\underset{\sim}{T}}\,\underset{\sim}{\phi}$, where, for corner nodes, $\underset{\sim}{\psi}$ has identical entries to the potential vector $\underset{\sim}{\phi}$, but for an FCSN the $\underset{\sim}{\psi}$ entry is the difference between its electrostatic potential and the average of that of its corners. In terms of $\underset{\sim}{\psi}$, corner node interpolation functions are constructed neglecting the FCSN's, and FCSN interpolation functions are unity at the FCSN and zero on all other faces. In terms of these interpolation functions, the matrices $\underset{\sim}{\underset{\sim}{W}}$ and $\underset{\sim}{\underset{\sim}{V}}$ are defined by

$$\int |\underset{\sim}{\nabla}\phi|^2 \, d^3\underset{\sim}{r} = \underset{\sim}{\phi}^T \underset{\sim}{\underset{\sim}{W}}\,\underset{\sim}{\phi} = \underset{\sim}{\psi}^T \underset{\sim}{\underset{\sim}{A}}\,\underset{\sim}{\psi}$$

$$\int \phi^2 \, d^3\underset{\sim}{r} = \underset{\sim}{\phi}^T \underset{\sim}{\underset{\sim}{V}}\,\underset{\sim}{\phi} = \underset{\sim}{\psi}^T \underset{\sim}{\underset{\sim}{B}}\,\underset{\sim}{\psi}$$

where the integrals are over the element volume, and we have defined $\underset{\sim}{\underset{\sim}{A}}$ and $\underset{\sim}{\underset{\sim}{B}}$, which are readily shown to be given by

$$A_{ij} = \int \underset{\sim}{\nabla} N^i \cdot \underset{\sim}{\nabla} N^j \, d^3\underset{\sim}{r}$$

$$B_{ij} = \int N^i \, N^j \, d^3\underset{\sim}{r} \quad .$$

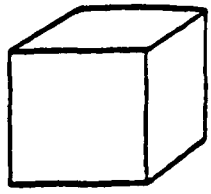
Finally,

$$\underset{\sim}{\underset{\sim}{W}} = \underset{\sim}{\underset{\sim}{T}}^T \underset{\sim}{\underset{\sim}{A}}\,\underset{\sim}{\underset{\sim}{T}}$$

$$\underset{\sim}{\underset{\sim}{V}} = \underset{\sim}{\underset{\sim}{T}}^T \underset{\sim}{\underset{\sim}{B}}\,\underset{\sim}{\underset{\sim}{T}} \quad .$$

The matrices $\underset{\sim}{\underset{\sim}{T}}$, $\underset{\sim}{\underset{\sim}{W}}$ and $\underset{\sim}{\underset{\sim}{V}}$ are given for each element type in the succeeding sections.

4.21.22 THE EMPTY CUBE (TYPE 0) ELEMENT WITH NO FCSN'S



Standard Cell 0
Empty trilinear cube
Orientation: Arbitrary
Potential Function:

i	N^i
1	(1-x)(1-y)(1-z)
2	(1-z)(1-y)x
3	(1-x)y(1-z)
4	(1-z)yx
5	z(1-y)(1-x)
6	x(1-y)(z)
7	zy(1-x)
8	xyz

W_{iJ}

1/3							
0	1/3						
0	-1/12	1/3					
-1/12	0	0	1/3				
0	-1/12	-1/12	-1/12	1/3			
-1/12	0	-1/12	-1/12	0	1/3		
-1/12	-1/12	0	-1/12	0	-1/12	1/3	
-1/12	-1/12	-1/12	0	-1/12	0	0	1/3

V_{ij}

1/27							
1/54	1/27						
1/54	1/108	1/27					
1/108	1/54	1/54	1/27				
1/54	1/108	1/108	1/216	1/27			
1/108	1/54	1/216	1/108	1/54	1/27		
1/108	1/216	1/54	1/108	1/54	1/108	1/27	
1/216	1/108	1/108	1/54	1/108	1/54	1/54	1/27

4.21.23 THE EMPTY CUBE (TYPE 0) ELEMENT WITH SIX FCSN'S

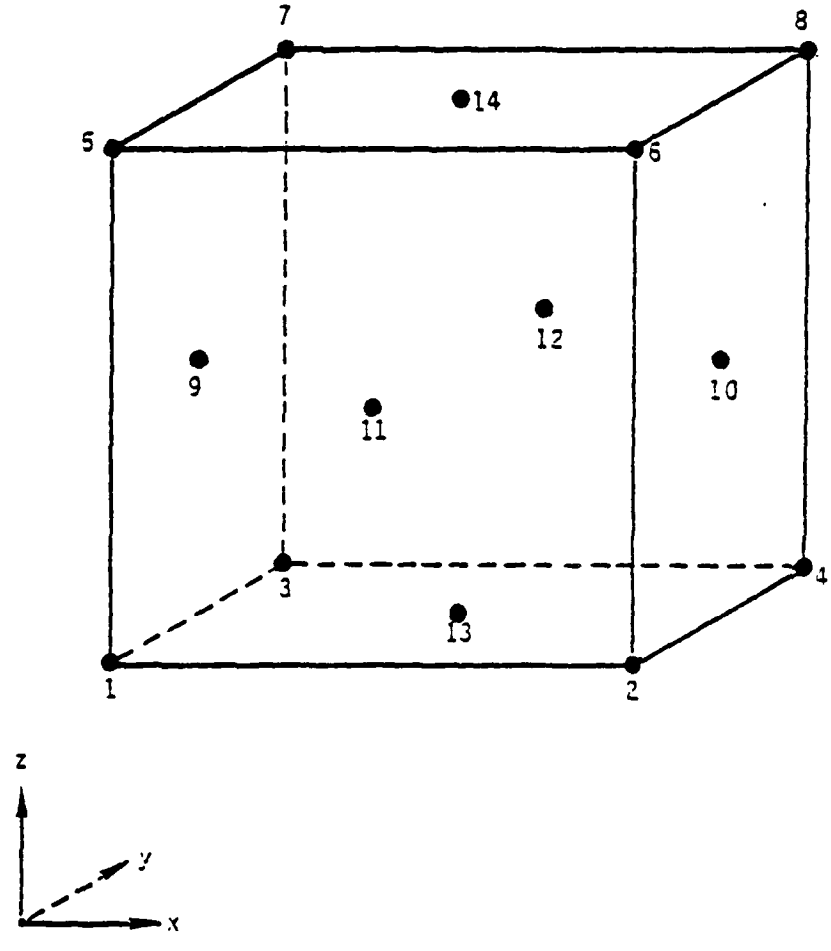


Figure 4.21.23/1. Cubical finite elements with six face-centered surface nodes (FCSNs). The FCSNs are located on the $\mp x$, $\mp y$, $\mp z$ faces respectively.

$$\phi(\underset{\sim}{r}) = \sum_i N^i \, \underset{\sim}{\psi}_i$$

$0 < x < 1;\ \bar{x} = 1 - x$

$0 < y < 1;\ \bar{y} = 1 - y$

$0 < z < 1;\ \bar{z} = 1 - z$

i	N^i	i	N^i
1	$\bar{x}\,\bar{y}\,\bar{z}$	9	$16\,\bar{x}\,y\,\bar{y}\,z\,\bar{z}$
2	$x\,\bar{y}\,\bar{z}$	10	$16\,x\,y\,\bar{y}\,z\,\bar{z}$
3	$\bar{x}\,y\,\bar{z}$	11	$16\,x\,\bar{x}\,\bar{y}\,z\,\bar{z}$
4	$x\,y\,\bar{z}$	12	$16\,x\,\bar{x}\,y\,z\,\bar{z}$
5	$\bar{x}\,\bar{y}\,z$	13	$16\,x\,\bar{x}\,y\,\bar{y}\,\bar{z}$
6	$x\,\bar{y}\,z$	14	$16\,x\,\bar{x}\,y\,\bar{y}\,z$
7	$\bar{x}\,y\,z$		
8	$x\,y\,z$		

Figure 4.21.23/2. Interpolation functions for the cubical element of Figure 4.21.23/1.

The Matrix T

1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
-.25	0.00	-.25	0.00	-.25	0.00	-.25	0.00	1.00	0.00	0.00	0.00	0.00	0.00
0.00	-.25	0.00	-.25	0.00	-.25	0.00	-.25	0.00	1.00	0.00	0.00	0.00	0.00
-.25	-.25	0.00	0.00	-.25	-.25	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
0.00	0.00	.25	-.25	0.00	0.00	-.25	-.25	0.00	0.00	0.00	1.00	0.00	0.00
-.25	-.25	-.25	-.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00	-.25	-.25	-.25	-.25	0.00	0.00	0.00	0.00	0.00	1.00

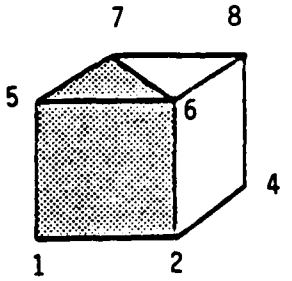
The Matrix W

+.79778	+.48074	+.48074	+.41370	+.48074	+.41370	+.41370	+.43000	-.73037	-.57333	-.73037	-.57333	-.73037	-.57333
+.48074	+.79778	+.41370	+.48074	+.41370	+.48074	+.43000	+.41370	-.57333	-.73037	-.73037	-.57333	-.73037	-.57333
+.48074	+.41370	+.79778	+.48074	+.41370	+.43000	+.48074	+.41370	-.73037	-.57333	-.57333	-.73037	-.73037	-.57333
+.41370	+.48074	+.48074	+.79778	+.43000	+.41370	+.41370	+.48074	-.57333	-.73037	-.57333	-.73037	-.73037	-.57333
+.48074	+.41370	+.41370	+.43000	+.79778	+.48074	+.48074	+.41370	-.73037	-.57333	-.73037	-.57333	-.57333	-.73037
+.41370	+.48074	+.43000	+.41370	+.48074	+.79778	+.41370	+.48074	-.57333	-.73037	-.73037	-.57333	-.57333	-.73037
+.41370	+.43000	+.48074	+.41370	+.48074	+.41370	+.79778	+.48074	-.73037	-.57333	-.57333	-.73037	-.57333	-.73037
+.43000	+.41370	+.41370	+.48074	+.41370	+.48074	+.48074	+.79778	-.57333	-.73037	-.57333	-.73037	-.57333	-.73037
-.73037	-.57333	-.73037	-.57333	-.73037	-.57333	-.73037	-.57333	2.18074	+.66370	+.59259	+.59259	+.59259	+.59259
-.57333	-.73037	-.57333	-.73037	-.57333	-.73037	-.57333	-.73037	+.66370	2.18074	+.59259	+.59259	+.59259	+.59259
-.73037	-.73037	-.57333	-.57333	-.73037	-.73037	-.57333	-.57333	+.59259	+.59259	2.18074	+.66370	+.59259	+.59259
-.57333	-.57333	-.73037	-.73037	-.57333	-.57333	-.73037	-.73037	+.59259	+.59259	+.66370	2.18074	+.59259	+.59259
-.73037	-.73037	-.73037	-.73037	-.57333	-.57333	-.57333	-.57333	+.59259	+.59259	+.59259	+.59259	2.18074	+.66370
-.57333	-.57333	-.57333	-.57333	-.73037	-.73037	-.73037	-.73037	+.59259	+.59259	+.59259	+.59259	+.66370	2.18074

The Matrix V

+.02148	+.00926	+.00926	+.00630	+.00926	+.00630	+.00630	+.00796	−.01630	−.02296	−.01630	−.02296	−.01630	−.02296
+.00926	+.02148	+.00630	+.00926	+.00630	+.00926	+.00796	+.00630	−.02296	−.01630	−.01630	−.02296	−.01630	−.02296
+.00926	+.00630	+.02148	+.00926	+.00630	+.00796	+.00926	+.00630	−.01630	−.02296	−.02296	−.01630	−.01630	−.02296
+.00630	+.00926	+.00926	+.02148	+.00796	+.00630	+.00630	+.00926	−.02296	−.01630	−.02296	−.01630	−.01630	−.02296
+.00926	+.00630	+.00630	+.00796	+.02148	+.00926	+.00926	+.00630	−.01630	−.02296	−.01630	−.02296	−.02296	−.01630
+.00630	+.00926	+.00796	+.00630	+.00926	+.02148	+.00630	+.00926	−.02296	−.01630	−.01630	−.02296	−.02296	−.01630
+.00630	+.00796	+.00926	+.00630	+.00926	+.00630	+.02148	+.00926	−.01630	−.02296	−.02296	−.01630	−.02296	−.01630
+.00796	+.00630	+.00630	+.00926	+.00630	+.00926	+.00926	+.02148	−.02296	−.01630	−.02296	−.01630	−.02296	−.01630
−.01630	−.02296	−.01630	−.02296	−.01630	−.02296	−.01630	−.02296	+.09481	+.04741	+.05926	+.05926	+.05926	+.05926
−.02296	−.01630	−.02296	−.01630	−.02296	−.01630	−.02296	−.01630	+.04741	+.09481	+.05926	+.05926	+.05926	+.05926
−.01630	−.01630	−.02296	−.02296	−.01630	−.01630	−.02296	−.02296	+.05926	+.05926	+.09481	+.04741	+.05926	+.05926
−.02296	−.02296	−.01630	−.01630	−.02296	−.02296	−.01630	−.01630	+.05926	+.05926	+.04741	+.09481	+.05926	+.05926
−.01630	−.01630	−.01630	−.01630	−.02296	−.02296	−.02296	−.02296	+.05926	+.05926	+.05926	+.05926	+.09481	+.04741
−.02296	−.02296	−.02296	−.02296	−.01630	−.01630	−.01630	−.01630	+.05926	+.05926	+.05926	+.05926	+.04741	+.09481

4.21.24 THE WEDGE ELEMENT (TYPE 1)



Half Empty Wedge

$1 < x+y < 2$

$0 < z < 1$

Node	Location			Interpolation Function, N_i
1	0	0	0	0
2	1	0	0	$\bar{y}\,\bar{z}$
3	0	1	0	$\bar{x}\,\bar{z}$
4	1	1	0	$(x-\bar{y})\bar{z}$
5	0	0	1	0
6	1	0	1	$\bar{y}\,z$
7	0	1	1	$\bar{x}\,z$
8	1	1	1	$(x-\bar{y})z$
9	2/3	2/3	0	$27\bar{x}\,\bar{y}\,\bar{z}(x-\bar{y})$
10	2/3	2/3	1	$27\bar{x}\,\bar{y}\,z(x-\bar{y})$
11	1	1/2	1/2	$16(x-\bar{y})\bar{y}\,z\,z$
12	1/2	1	1/2	$16(x-\bar{y})\bar{x}\,z\,z$
13	1/2	1/2	1/2	$16\bar{x}\,\bar{y}\,z\,\bar{z}$

where

$\bar{x} = 1-x$

$\bar{y} = 1-y$

$\bar{z} = 1-z$

The Matrix T

for Type 1 (Wedge) Element

1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000
.0000	-.3333	-.3333	-.3333	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	-.3333	-.3333	-.3333	.0000	1.0000	.0000	.0000	.0000
.0000	-.2500	.0000	-.2500	.0000	-.2500	.0000	-.2500	.0000	.0000	1.0000	.0000	.0000
.0000	.0000	-.2500	-.2500	.0000	.0000	-.2500	-.2500	.0000	.0000	.0000	1.0000	.0000
.0000	-.2500	-.2500	.0000	.0000	-.2500	-.2500	.0000	.0000	.0000	.0000	.0000	1.0000

The Matrix W

for Type 1 (Wedge) Element

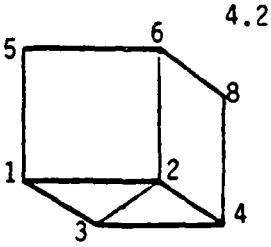
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.6813	0.4804	0.4748	0.0000	0.3520	0.3178	0.3958	-1.0189	-0.6311	-0.5556	-0.3630	-0.1333
0.0000	0.4804	0.6813	0.4748	0.0000	0.3178	0.3520	0.3956	-1.0189	-0.6311	-0.3630	-0.5556	-0.1333
0.0000	0.4748	0.4748	0.9257	0.0000	0.3956	0.3956	0.5132	-1.1689	-0.7811	-0.5111	-0.5111	-0.2074
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.3520	0.3178	0.3956	0.0000	0.6813	0.4804	0.4748	-0.6311	-1.0189	-0.5556	-0.3630	-0.1333
0.0000	0.3178	0.3520	0.3956	0.0000	0.4804	0.6813	0.4748	-0.6311	-1.0189	-0.3630	-0.5556	-0.1333
0.0000	0.3956	0.3956	0.5132	0.0000	0.4748	0.4748	0.9257	-0.7811	-1.1689	-0.5111	-0.5111	-0.2074
0.0000	-1.0189	-1.0189	-1.1689	0.0000	-0.6311	-0.6311	-0.7811	2.8316	1.2184	0.6000	0.6000	0.0000
0.0000	-0.6311	-0.6311	-0.7811	0.0000	-1.0189	-1.0189	-1.1689	1.2184	2.8316	0.6000	0.6000	0.0000
0.0000	-0.5556	-0.3630	-0.5111	0.0000	-0.5556	-0.3630	-0.5111	0.6000	0.6000	1.8963	0.2370	-0.4741
0.0000	-0.3630	-0.5556	-0.5111	0.0000	-0.3630	-0.5556	-0.5111	0.6000	0.6000	0.2370	1.8963	-0.4741
0.0000	-0.1333	-0.1333	-0.2074	0.0000	-0.1333	-0.1333	-0.2074	0.0000	0.0000	-0.4741	-0.4741	1.8963

The Matrix V

for Type 1 (Wedge) Element

0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0121	0.0023	0.0021	0.0000	0.0041	0.0012	0.0011	-0.0038	-0.0089	-0.0047	-0.0099	-0.0052
0.0000	0.0023	0.0121	0.0021	0.0000	0.0012	0.0041	0.0011	-0.0038	-0.0089	-0.0099	-0.0047	-0.0052
0.0000	0.0021	0.0021	0.0118	0.0000	0.0011	0.0011	0.0038	-0.0034	-0.0086	-0.0047	-0.0047	-0.0104
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0300	0.0041	0.0012	0.0011	0.0000	0.0121	0.0023	0.0021	-0.0089	-0.0038	-0.0047	-0.0099	-0.0052
0.0000	0.0012	0.0041	0.0011	0.0000	0.0023	0.0121	0.0021	-0.0089	-0.0038	-0.0099	-0.0047	-0.0052
0.0000	0.0011	0.0011	0.0038	0.0000	0.0021	0.0021	0.0118	-0.0086	-0.0034	-0.0047	-0.0047	-0.0104
0.0000	-0.0038	-0.0038	-0.0034	0.0000	-0.0089	-0.0089	-0.0086	0.0439	0.0219	0.0275	0.0275	0.0290
0.0000	-0.0089	-0.0089	-0.0086	0.0000	-0.0038	-0.0038	-0.0034	0.0219	0.0439	0.0275	0.0275	0.0290
0.0000	-0.0047	-0.0099	-0.0047	0.0000	-0.0047	-0.0099	-0.0047	0.0275	0.0275	0.0474	0.0237	0.0237
0.0000	-0.0099	-0.0047	-0.0047	0.0000	-0.0099	-0.0047	-0.0047	0.0275	0.0275	0.0237	0.0474	0.0237
0.0000	-0.0052	-0.0052	-0.0104	0.0000	-0.0052	-0.0052	-0.0104	0.0290	0.0290	0.0237	0.0237	0.0474

4.21.25 THE TYPE 2 ELEMENT



Cube with Diagonal Line on One Face
Orientation: Line from 2 to 3

Node	Location			Interpolation Function, N_i
1	0	0	0	$(\bar{x}-y)\bar{z}\ominus(\bar{x}-y)$
2	1	0	0	$[x\ominus(\bar{x}-y) + \bar{y}\ominus(x-\bar{y})]\bar{z}$
3	0	1	0	$[y\ominus(\bar{x}-y) + \bar{x}\ominus(x-y)]\bar{z}$
4	1	1	0	$(x-\bar{y})\bar{z}\ominus(x-\bar{y})$
5	0	0	1	$\bar{x}\ \bar{y}\ z$
6	1	0	1	$x\ \bar{y}\ z$
7	0	1	1	$\bar{x}\ y\ z$
8	1	1	1	$x\ y\ z$
9	0	1/2	1/2	$16\ \bar{x}\ \bar{y}\ y\ z\ \bar{z}$
10	1	1/2	1/2	$16\ x\ y\ \bar{y}\ z\ \bar{z}$
11	1/2	0	1/2	$16\ x\ \bar{x}\ \bar{y}\ z\ \bar{z}$
12	1/2	1	1/2	$16\ x\ \bar{x}\ y\ z\ \bar{z}$
13	1/2	1/2	1	$16\ x\ \bar{x}\ y\ \bar{y}\ z$
14	1/3	1/3	0	$27\ x\ y\ \bar{z}(\bar{x}-y)\ominus(\bar{x}-y)$
15	2/3	2/3	0	$27\ \bar{x}\ \bar{y}\ \bar{z}(x-\bar{y})\ominus(x-\bar{y})$

where

$\bar{x} = 1-x$
$\bar{y} = 1-y$
$\bar{z} = 1-z$

The Matrix T

for Type 2 (Special Empty Cube) Element

1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
-.2500	.0000	-.2500	.0000	-.2500	.0000	-.2500	.0000	1.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	-.2500	.0000	-.2500	.0000	-.2500	.0000	-.2500	.0000	1.0000	.0000	.0000	.0000	.0000	.0000
-.2500	-.2500	.0000	.0000	-.2500	-.2500	.0000	.0000	.0000	.0000	1.0000	.0000	.0000	.0000	.0000
.0000	.0000	-.2500	-.2500	.0000	.0000	-.2500	-.2500	.0000	.0000	.0000	1.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	-.2500	-.2500	-.2500	-.2500	.0000	.0000	.0000	.0000	1.0000	.0000	.0000
-.3333	-.3333	-.3333	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	1.0000	.0000
.0000	-.3333	-.3333	-.3333	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	1.0000

The Matrix W

for Type 2 (Special Empty Cube) Element

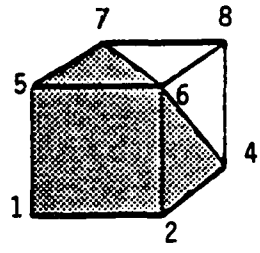
1.1617	0.5415	0.5415	0.4455	0.5708	0.4749	0.4749	0.4608	-0.8278	-0.8021	-0.8278	-0.8021	-0.8438	-1.0889	-0.1000
0.5415	1.3519	0.9714	0.5415	0.3589	0.4184	0.3713	0.3599	-0.5177	-0.8875	-0.8875	-0.5177	-0.5077	-1.0189	-1.0189
0.5415	0.9714	1.3519	0.5415	0.3598	0.3713	0.4184	0.3598	-0.8875	-0.5177	-0.5177	-0.8875	-0.5077	-1.0189	-1.0189
0.4455	0.5415	0.5415	1.1617	0.4608	0.4749	0.4749	0.5708	-0.8021	-0.8278	-0.8021	-0.8278	-0.8438	-0.1000	-1.0889
0.5708	0.3598	0.3599	0.4608	0.7954	0.4788	0.4788	0.4105	-0.7313	-0.5724	-0.7313	-0.5724	-0.7208	-0.3809	-0.2059
0.4749	0.4184	0.3713	0.4749	0.4788	0.7949	0.4144	0.4788	-0.5770	-0.7287	-0.7287	-0.5770	-0.7243	-0.2874	-0.2874
0.4749	0.3713	0.4184	0.4749	0.4788	0.4144	0.7949	0.4788	-0.7287	-0.5770	-0.5770	-0.7287	-0.7243	-0.2874	-0.2874
0.4608	0.3599	0.3599	0.5708	0.4105	0.4788	0.4788	0.7954	-0.5724	-0.7313	-0.5724	-0.7313	-0.7208	-0.2059	-0.3809
-0.8278	-0.5177	-0.8875	-0.8021	-0.7313	-0.5770	-0.7287	-0.5724	2.1752	0.8692	0.8017	0.5835	0.5928	0.4000	0.2000
-0.8021	-0.8875	-0.5177	-0.8278	-0.5724	-0.7287	-0.5770	-0.7313	0.8692	2.1752	0.5835	0.8017	0.5928	0.2000	0.4000
-0.8278	-0.8875	-0.5177	-0.8021	-0.7313	-0.7287	-0.5770	-0.5724	0.8017	0.5835	2.1752	0.8692	0.5928	0.4000	0.2000
-0.8021	-0.5177	-0.8875	-0.8278	-0.5724	-0.5770	-0.7287	-0.7313	0.5835	0.8017	0.8692	2.1752	0.5928	0.2000	0.4000
-0.8438	-0.5077	-0.5077	-0.8438	-0.7208	-0.7243	-0.7243	-0.7208	0.5928	0.5928	0.5928	0.5928	2.1498	0.3365	0.3365
-1.0889	-1.0189	-1.0189	-0.1000	-0.3809	-0.2874	-0.2874	-0.2059	0.4000	0.2000	0.4000	0.2000	0.3365	2.0316	0.0000
-0.1000	-1.0189	-1.0189	-1.0889	-0.2059	-0.2874	-0.2874	-0.3809	0.2000	0.4000	0.2000	0.4000	0.3365	0.0000	2.0316

The Matrix V

for Type 2 (Special Empty Cube) Element

0.0223	0.0047	0.0047	0.0118	0.0111	0.0083	0.0083	0.0103	-0.0258	-0.0251	-0.0258	-0.0251	-0.0275	-0.0090	-0.0089
0.0047	0.0217	0.0069	0.0047	0.0025	0.0064	0.0054	0.0025	-0.0162	-0.0088	-0.0088	-0.0162	-0.0187	-0.0038	-0.0038
0.0047	0.0069	0.0217	0.0047	0.0025	0.0054	0.0064	0.0025	-0.0088	-0.0162	-0.0162	-0.0088	-0.0187	-0.0038	-0.0038
0.0118	0.0047	0.0047	0.0223	0.0103	0.0083	0.0083	0.0111	-0.0251	-0.0258	-0.0251	-0.0258	-0.0275	-0.0089	-0.0090
0.0111	0.0025	0.0025	0.0103	0.0217	0.0097	0.0097	0.0084	-0.0168	-0.0230	-0.0186	-0.0230	-0.0172	-0.0086	-0.0106
0.0083	0.0064	0.0054	0.0083	0.0097	0.0220	0.0072	0.0097	-0.0235	-0.0161	-0.0161	-0.0235	-0.0183	-0.0106	-0.0106
0.0083	0.0054	0.0064	0.0083	0.0097	0.0072	0.0220	0.0097	-0.0161	-0.0235	-0.0235	-0.0161	-0.0183	-0.0106	-0.0106
0.0103	0.0025	0.0025	0.0111	0.0084	0.0097	0.0097	0.0217	-0.0230	-0.0168	-0.0230	-0.0168	-0.0172	-0.0106	-0.0086
-0.0256	-0.0162	-0.0088	-0.0251	-0.0168	-0.0235	-0.0161	-0.0230	0.0943	0.0480	0.0602	0.0584	0.0599	0.0387	0.0178
-0.0251	-0.0088	-0.0162	-0.0258	-0.0230	-0.0161	-0.0235	-0.0186	0.0480	0.0943	0.0584	0.0602	0.0599	0.0178	0.0387
-0.0258	-0.0088	-0.0162	-0.0251	-0.0168	-0.0161	-0.0235	-0.0230	0.0602	0.0584	0.0943	0.0480	0.0599	0.0387	0.0178
-0.0251	-0.0162	-0.0088	-0.0258	-0.0230	-0.0235	-0.0161	-0.0188	0.0584	0.0602	0.0480	0.0943	0.0599	0.0178	0.0387
-0.0275	-0.0167	-0.0167	-0.0275	-0.0172	-0.0183	-0.0183	-0.0172	0.0599	0.0599	0.0599	0.0599	0.0992	0.0213	0.0213
-0.0090	-0.0038	-0.0038	-0.0089	-0.0088	-0.0108	-0.0106	-0.0106	0.0387	0.0178	0.0387	0.0178	0.0213	0.0439	0.0000
-0.0089	-0.0038	-0.0038	-0.0090	-0.0108	-0.0108	-0.0106	-0.0088	0.0178	0.0387	0.0178	0.0387	0.0213	0.0000	0.0439

4.21.26 THE TETRAHEDRON ELEMENT



Tetrahedron (Type 3)

$2 < x+y+z < 3$

Node	Location			Volume	N_i
1	0	0	0	0	0
2	1	0	0	0	0
3	0	1	0	0	0
4	1	1	0	.007756	$1 - z$
5	0	0	1	0	0
6	1	0	1	.007756	$1 - y$
7	0	1	1	.007756	$1 - x$
8	1	1	1	.009542	$x + y + z - 2$
9	1	2/3	2/3	.032101	$27\, x\, \bar{y}\, \bar{z}(x + y + z - 2)$
10	2/3	1	2/3	.032101	$27\, \bar{x}\, y\, \bar{z}(x + y + z - 2)$
11	2/3	2/3	1	.032101	$27\, \bar{x}\, \bar{y}\, z(z - \bar{x} - \bar{y})$
12	2/3	2/3	2/3	.037552	$27\, \bar{x}\, \bar{y}\, \bar{z}$

The Matrix W

For the Type 3 (Tetrahedron) Volume Element

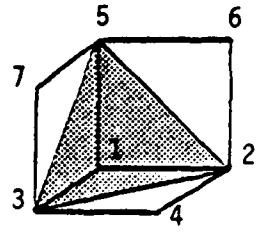
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	.2975	0	.1190	.1190	.1463	-.3254	-.3254	-.0657	.0346
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	.1190	0	.2975	.1190	.1463	-.3254	-.0657	-.3254	.0346
0	0	0	.1190	0	.1190	.2975	.1463	-.0657	-.3254	-.3254	.0346
0	0	0	.1463	0	.1463	.1463	.5836	-.3107	-.3107	-.3107	-.0905
0	0	0	-.3254	0	-.3254	-.0657	-.3107	1.5055	.0539	.0539	-.5861
0	0	0	-.3254	0	-.0657	-.3254	-.3107	.0539	1.5055	.0539	-.5861
0	0	0	-.0657	0	-.3254	-.3254	-.3107	.0539	.0539	1.5055	-.5861
0	0	0	.0346	0	.0346	.0346	-.0905	-.5861	-.5861	-.5861	1.7451

The Matrix V

For the Type 3 (Tetrahedron) Volume Element

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	.0058	0	.0001	.0001	.0001	.0011	.0011	-.0015	.0011
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	.0001	0	.0058	.0001	.0001	.0011	-.0015	.0011	.0011
0	0	0	.0001	0	.0001	.0058	.0001	-.0015	.0011	.0011	.0011
0	0	0	.0001	0	.0001	.0001	.0057	.0016	.0016	.0016	-.0010
0	0	0	.0011	0	.0011	-.0015	.0016	.0132	.0051	.0051	.0063
0	0	0	.0011	0	-.0015	.0011	.0016	.0051	.0132	.0051	.0063
0	0	0	-.0015	0	.0011	.0011	.0016	.0051	.0051	.0132	.0063
0	0	0	.0011	0	.0011	.0011	-.0010	.0063	.0063	.0063	.0163

4.21.27 THE TRUNCATED CUBE ELEMENT (TYPE 4)



Truncated Cube (Type 4)

$1 < x+y+z < 3$

Node	Location			Volume	N_i
1	0	0	0	0	0
2	1	0	0	-.0380	$(2x - y - z + 1)\ K(2 - x - y - z)/3$
3	0	1	0	-.0380	$(2y - x - z + 1)\ K(2 - x - y - z)/3$
4	1	1	0	.0406	$(1 + x + y - 2z)\ \theta_2/3. + (1-z)\theta_3$
5	0	0	1	-.0380	$(2z - x - y + 1)\ K(2 - x - y - z)/3$
6	1	0	1	.0406	$(1 + x + z - 2y)\ \theta_2/3 + (1-y)\theta_3$
7	0	1	1	.0406	$(1 + y + z - 2x)\ \theta_2/3 + (1-x)\theta_3$
8	1	1	1	-.0709	$(x + y + z - 2)\ \theta_3$
9	0	2/3	2/3	.1119	$27\ \bar{x}\ \bar{y}\ \bar{z}\ K(y + z - 1)$
10	1	1/2	1/2	.1499	$16\ x\ y\ \bar{y}\ z\ \bar{z}(x + y + z - 1)$
11	2/3	0	2/3	.1119	$27\ \bar{x}\ \bar{y}\ \bar{z}\ K(x + z - 1)$
12	1/2	1	1/2	.1499	$16\ x\ \bar{x}\ y\ z\ \bar{z}(x + y + z - 1)$
13	2/3	2/3	0	.1119	$27\ \bar{x}\ \bar{y}\ \bar{z}\ K(x + y - 1)$
14	1/2	1/2	1	.1499	$16\ x\ \bar{x}\ y\ \bar{y}\ z(x + y + z - 1)$
15	1/3	1/3	1/3	.1112	$K(1 - x - y + 2z)\ K(1 - y - z + 2x)$ $K(1 - x - z + 2y)\ K(2 - x - y - z)$

$K(s) = s\ _s(s)$

$\theta_2 = (x + y + z - 1)\ \ (2 - x - y - z)$

$\theta_3 = (x + y + z - 2)$

$\bar{x} = 1 - x$

$\bar{y} = 1 - y$

$\bar{z} = 1 - z$

The Matrix W

For the Type 4 (Truncated Cube) Volume Element

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	2.4078	1.7513	.1544	1.7513	.1544	.0788	.8121	-.4814	-1.1838	-.8123	-.8248	-.8123	-.8248	-2.3878
0	1.7513	2.4078	.1544	1.7513	.0788	.1544	.8121	-.8123	-.8248	-.4814	-1.1838	-.8123	-.8248	-2.3878
0	.1544	.1544	.7881	.0788	.1283	.1283	.1383	-.1854	-.0877	-.1854	-.0877	-.8588	-.2842	.2818
0	1.7513	1.7513	.0788	2.4078	.1544	.1544	.8121	-.8123	-.8248	-.8123	-.8248	-.4814	-1.1838	-2.3878
0	.1544	.0788	.1283	.1544	.7881	.1283	.1383	-.1854	-.0877	-.8588	-.2842	-.1854	-.0877	.2818
0	.0788	.1544	.1283	.1544	.1283	.7881	.1383	-.8588	-.2842	-.1854	-.0877	-.1854	-.0877	.2818
0	.8121	.8121	.1383	.8121	.1383	.1383	1.2278	-.4383	-.7551	-.4383	-.7551	-.4383	-.7551	-.4888
0	-.4814	-.8123	-.1854	-.8123	-.1854	-.8588	-.4383	2.4388	.3514	.3258	.2515	.3258	.2515	.1482
0	-1.1838	-.8248	-.0877	-.8248	-.0877	-.2842	-.7551	.3514	1.8845	.2515	.1374	.2515	.1374	.8848
0	-.8123	-.4814	-.1854	-.8123	-.8588	-.1854	-.4383	.3258	.2515	2.4388	.3514	.3258	.2515	.1482
0	-.8248	-1.1838	-.0877	-.8248	-.2842	-.0877	-.7551	.2515	.1374	.3514	1.8845	.2515	.1374	.8848
0	-.8123	-.8123	-.8588	-.4814	-.1854	-.1854	-.4383	.3258	.2515	.3258	.2515	2.4388	.3514	.1482
0	-.8248	-.8248	-.2842	-1.1838	-.0877	-.0877	-.7551	.2515	.1374	.2515	.1374	.3514	1.8845	.8848
0	-2.3878	-2.3878	.2818	-2.3878	.2818	.2818	-.4888	.1482	.8848	.1482	.8848	.1482	.8848	4.3884

The Matrix V

For the Type 4 (Truncated Cube) Volume Element

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	.0268	.0132	-.0002	.0132	-.0002	-.0010	.0121	-.0124	-.0283	-.0111	-.0207	-.0111	-.0207	-.0014
0	.0132	.0268	-.0002	.0132	-.0010	-.0002	.0121	-.0111	-.0207	-.0124	-.0283	-.0111	-.0207	-.0014
0	-.0002	-.0002	.0150	-.0010	.0012	.0012	-.0027	.0018	.0085	.0018	.0085	.0083	-.0013	.0020
0	.0132	.0132	-.0010	.0268	-.0002	-.0002	.0121	-.0111	-.0207	-.0111	-.0207	-.0124	-.0283	-.0014
0	-.0002	-.0010	.0012	-.0002	.0150	.0012	-.0027	.0018	.0085	.0083	-.0013	.0018	.0085	.0020
0	-.0010	-.0002	.0012	-.0002	.0012	.0150	-.0027	.0083	-.0013	.0018	.0085	.0018	.0085	.0020
0	.0121	.0121	-.0027	.0121	-.0027	-.0027	.0235	-.0159	-.0215	-.0159	-.0215	-.0159	-.0215	-.0103
0	-.0124	-.0111	.0018	-.0111	.0018	.0083	-.0159	.0482	.0188	.0101	.0287	.0101	.0287	.0101
0	-.0283	-.0207	.0085	-.0207	.0085	-.0013	-.0215	.0188	.0838	.0287	.0359	.0287	.0359	.0137
0	-.0111	-.0124	.0018	-.0111	.0083	.0018	-.0159	.0101	.0287	.0482	.0188	.0101	.0287	.0101
0	-.0207	-.0283	.0085	-.0207	-.0013	.0085	-.0215	.0287	.0359	.0188	.0838	.0287	.0359	.0137
0	-.0111	-.0111	.0083	-.0124	.0018	.0018	-.0159	.0101	.0287	.0101	.0287	.0482	.0188	.0101
0	-.0207	-.0207	-.0013	-.0283	.0085	.0085	-.0215	.0287	.0359	.0287	.0359	.0188	.0838	.0137
0	-.0014	-.0014	.0020	-.0014	.0020	.0020	-.0103	.0101	.0137	.0101	.0137	.0101	.0137	.0482

4.21.28 THE SLANTED THIN PLATE ELEMENT (TYPE 5)

The type 5 element is treated as two type 1 (q.v.) elements.

4.30 MATRIX SOLVERS

POLAR (NTERAK) currently uses two methods to solve matrix equations of the form

$$\underset{\sim}{M}\,\underset{\sim}{X} = \underset{\sim}{d}$$

where $\underset{\sim}{M}$ is a given matrix, $\underset{\sim}{d}$ is data and $\underset{\sim}{X}$ is the solution vector. These methods are the Conjugate Gradient Method and the Incomplete Cholesky Conjugate Gradient Method (ICCG).

4.31 CONJUGATE GRADIENT METHOD

This method is used for the Poisson equation solution (4.20, 4.44) where $\underset{\sim}{X} = \underset{\sim}{\phi}$ (potential) which can have 10,000 or more components. The next revision of this document will have a complete discussion of this method.

4.32 ICCG, THE INCOMPLETE CHOLESKY CONJUGATE GRADIENT METHOD

ICCG is used to solve the charging equations (4.50, 5.70), where $\underset{\sim}{X} = \underset{\sim}{V}_S$, surface voltages. Components of $\underset{\sim}{V}_S$ generally number 1000 or less. This allows the entire problem to be kept in the machine (5.73.2). ICCG will find series of approximate inverses for $\underset{\approx}{M}$ finding $\underset{\sim}{X}$ as

$$\underset{\sim}{X} = \underset{\approx}{M}^{-1} \underset{\sim}{d} .$$

It is iterative, but iterates on $\underset{\approx}{M}^{-1}$ as well as $\underset{\sim}{X}$. A more complete description will be found here in future revisions. Reference: Kershaw, D. (1978), "The Incomplete Cholesky-Conjugate Gradient Method for the Iterative Solution of Systems of Linear Equations," *Journal of Computational Physics*, 26, p. 43.

4.40 SPACE CHARGE AND CURRENT COMPUTATION

This section describes the numerical techniques such as integration, that are used to effect the models described in Chapter 3. In some cases, the computational requirements are satisfied by the top-down structuring of simple subroutines. In these cases, the Chapter 4 discussion is deferred to Chapter 5 to avoid repetition.

4.41 WEAK FIELD IONS, PRESHEATH AND WAKE

This model, presented in Section 3.30, will be elaborated upon here in future revisions. The reader is currently referred to Section 5.61.

4.42 THE POLAR SHEATH MODEL (TECHNICAL)

Section 3.60 contains a general discussion of the POLAR sheath model. This section is not designed to be a complete discussion, but to fill in the technical details of the model.

4.42.1 ION SHEATH EDGE ALGORITHM (SHEATH)

The SHEATH routine takes as input a sheath edge potential, PMIN, and the eight vertex potentials of a single cubic element, P(2,2,2). It determines whether the sheath potential contour passes through the specified element and if it does it generates a number of triangles, NPART, with areas W(I) and center X(3,I) which approximates the equipotential surface.

The sheath location proceeds by finding any edges whose vertex potentials bracket the sheath potential. If none do, NPART is set to zero and control returns to the calling program. For the cases of three intersections a single triangle is constructed from the intersection points with the area calculated by the TRiangle AREA routine. For four or five edge intersections the centroid of the intersection points is found and triangles constructed using adjacent intersection points and the centroid. Thus for four edge intersections, four triangles are formed. SHEATH then would return NPART = 4 and the center coordinates of each of the four triangles.

4.42.2 ION CURRENTS TO THE SHEATH SURFACE

We calculate the current density to the sheath edge by assuming it to be a perfectly absorbing spherical surface in a flowing plasma. We assume that the potential around the sphere is spherically symmetric and attracts ions. We utilize the coordinate systems

indicated in Figure 3.42/1 below (shown twice to reduce cluttering).

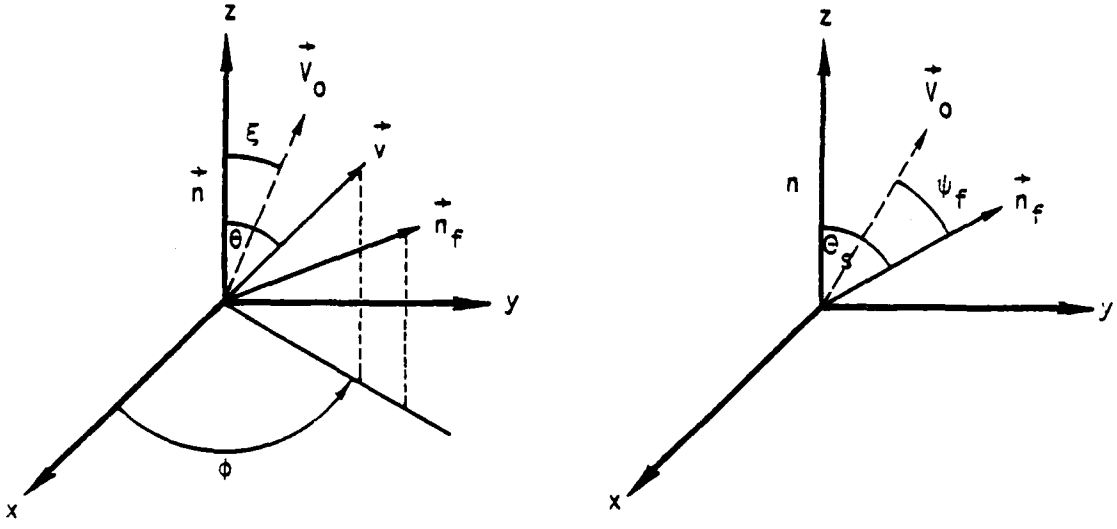


Figure 4.42/1.

and introduce the following definitions:

$\vec{a}$ = sphere radius

V_o = satellite velocity

$\Phi(\vec{r})$ = potential energy of ion at $\vec{r}$

$\vec{n}$ = unit vector at position r on sphere where normal current density is to be calculated

$\vec{n}_f$ = unit vector in final direction (at r =) mapped by particles launched from $\vec{r} \equiv (a,\vec{n})$ with velocity v

x-z plane = plane determined by n and V_o

ξ = angle between $\vec{n}$ and $\vec{V}_0$

Θ_∞ = polar angle of particle for $(n = \infty, n_f)$

ϕ = angle between x-z plane and orbital plane

$f_0(\vec{V}_0) = (2\pi V_T^2)^{-3/2} \left\{ \exp -(\vec{v}_0 - \vec{V}_0)^2/2v_T^2 \right\}$

m = ion mass

v_T^2 = kT/m

For a particle moving in a central potential the conserved quantities are (e = m = 1):

$$\frac{1}{2} v^2 + \Phi(a) = \frac{1}{2} v_0^2 = E \qquad \text{Energy}$$

$$L = va\sin\theta \qquad \text{Angular Momentum}$$

$$\phi \qquad \text{Azimuth}$$

The normal current density $j(\vec{r})$ at a point $\vec{r} = (a,\vec{n})$ on the sphere is given by

$$j = j(a,\vec{n}) = \int (\vec{v}\cdot\vec{n})\, f_0(\vec{v}_0)\, d^3\vec{v}$$

$$= (2\pi v_T^2)^{-3/2} \exp \frac{1}{2} (V_0^2 - 2V_0 v_0 \cos\psi_f + V_0^2)/v_T^2$$

$$\times v^3 \cos\theta \sin\theta \, dv \, d\theta \, d\phi$$

where

$$\cos\psi_f = \cos\xi \cos\theta_\infty - \sin\xi \sin\theta_\infty \cos\phi$$

$$\theta_\infty(v_0,\theta) = a \sin\theta \int_a^\infty \frac{dr}{r^2} \left[1 - \frac{a^2}{r^2} \sin^2\theta + \frac{\Phi(a) - \Phi(r)}{\frac{1}{2} v_0^2 - \Phi(a)} \right]^{-1/2}$$

Performing the ϕ integration, which can be done analytically, and using energy conservation,

$$j = (2\pi v_T^2)^{-3/2} \int_0^\infty v_o(v_o^2 - 2\phi(a))\, e^{-v_o^2/2v_T^2} F(v_o)\, dv_o$$

$$F(v_o) = 2\pi \int_0^{\theta_{max}(v_o)} e^{-v_o V \cos\xi \cos\theta_s / v_T^2} \quad e^{-V_o^2/2v_T^2}$$

$$\cdot I_o\left(\frac{v_o V_o \sin\xi \sin\theta_s}{v_T^2}\right) \quad \sin\rho \cos\theta \, d\theta$$

where I_o is the modified Bessel function of zero order.

For numerical calculations we define $t = x/3.75$ and approximate I_o by

$$I(x) = 1 + 3.5156229t^2 + 3.0899424t^4 + 1.2067492t^6 + .2659732t^8 + .0360768t^{10} + .0045813t^{12} + \epsilon$$

$$\epsilon < 1.6 \times 10^{-7}$$

for $-3.75 \leq x \leq 3.75$

$$x^{1/2} e^{-x} I_o(x) = .39894228 + .01328592\, t^{-1} + .00225319t^{-2} - .00157565t^{-3} + .00916281t^{-4} - .02057706t^{-5} + .02635537^{-6} - .01647633t^{-77} + .00392377t^{-8} + \epsilon$$

$$\epsilon < 1{,}9 \times 10^{-7}$$

for $3.75 \leq x < \infty$

For an inverse square potential

$\theta_{max} = \pi/2$, and the change in polar angle can be found analytically

$$\theta_s = \frac{\sin\theta}{(\sin^2\theta - b^2)^{1/2}} \sin^{-1}\left(\frac{\sin^2\theta - b^2}{1 - b^2}\right)^{1/2} ; \sin^2\theta - b^2 \geq 0$$

$$= \frac{\sin\theta}{(b^2 - \sin^2\theta)^{1/2}} \sinh^{-1}\left(\frac{b^2 - \sin^2\theta}{1 - b^2}\right)^{1/2} ; \sin^2\theta - b^2 < 0$$

where

$$b^2 = \frac{|\varphi(a)|}{E + |\varphi(a)|} \leq 1 \qquad (0 \leq E < \infty)$$

$$E = \frac{1}{2} v_o^2$$

$$\sinh^{-1} x = \ell n \; x + \sqrt{x^2 + 1}$$

Note for $1 - b^2 << 1$, θ_s may be $>2\pi$, i.e., the particles may execute a spiralling orbit. Accuracy may require that δv_o, $\delta\theta$ not produce large $\delta\theta_s$.

POLAR is currently using a temporary approximate model for initial velocities that is described here. This model will be replaced when the new model is ready. Initial velocities are approximated by adding a plasma rest frame characteristic velocity to the Mach velocity, $\vec{V}_M$ (spacecraft velocity normalized by the square root of (kT/m).) The characteristic velocity, $\vec{V}_c$, is constructed by first realizing that inside the sheath edge the density contribution of a trajectory is simply a continuity calculation (convergence effects come from the convergence of trajectories). Thus we estimate $|\vec{V}_c|$ by dividing the orbit-limited flux obtained for a totally absorbing surface (Laframboise and Parker, ref. 4-4) by a simplified form for the orbit-limited density (accurate in the low potential limit),

$$V_c = N_o \sqrt{\frac{kT}{2\pi m}} (1+\Phi) \Big/ \frac{N_o}{\sqrt{\pi}} \left(\sqrt{\Phi} + \frac{\sqrt{\pi}}{2}\right)$$

$$= \frac{(1+\Phi)}{\sqrt{2}\left(\sqrt{\Phi} + \frac{\sqrt{\pi}}{2}\right)} \cdot \left(\sqrt{\frac{kT}{m}}\right)$$

where $\Phi = eV/kT$, and the ion acoustic speed $\sqrt{kT/m}$ has been isolated to indicate the usual POLAR velocity normalization. The direction assigned to $\vec{V}_c$ is the normalized inward E field

$$\vec{V}_c = V_c \cdot \hat{E}$$

In the absence of flow, this result is exactly the result that would be obtained from the calculation of the velocity and density moments of the orbit-limited distribution function.

Our present (temporary) treatment of flow is more approximate. The initial sheath particle velocity, $\vec{V}_s$, is

$$\vec{V}_s = \begin{cases} \vec{V}_M + \vec{V}_C & \text{if } \left|(\vec{V}_M + \vec{V}_C)\right| \cdot \vec{E} > 0 \\ \text{or} & \\ \vec{V}_C & \text{if } \left|(\vec{V}_M + \vec{V}_C)\right| \cdot \vec{E} < 0 \end{cases}$$

where the conditional insures that $\vec{V}_s$ is always directed into the sheath.

4.42.3 SHEATH PARTICLE ASSIGNMENT

Each volume element of a problem is inspected for the presence of the sheath edge equipotential as described in Section 4.42.1. Each triangular subsurface of the sheath is a potential sheath particle; however, many times a portion of an equipotential is not really a source of current. This is assumed to occur when there exists just "outside" the sheath (in the anti E direction) a portion of the object, or a high potential region of the opposite sign. Both of these conditions are checked by calculating the inward initial velocity for the particle as per Section 4.42.2, reversing it, and tracking the particle backwards through two volume elements. If no "obstacles" are found, the particle is assigned a current or weight and placed in a particle list. This list is later read and the trajectories advanced as described in Sections 4.42.4 and 5.62. Finally, the sheath currents and velocities are not calculated for each particle but interpolated from a pre-calculated table of values.

4.42.4 TRAJECTORY TRACKING

Trajectory tracking can be an expensive endeavor, and a source of unpredictable error. To combat these problems, POLAR uses two different methods to follow ion trajectories (presently, there is no tracking of electrons).

The full step method is used in empty elements that do not touch the object where complex $\vec{E}$ fields are not anticipated. For these elements, $\vec{E}$ at the cell center is used for the entire cell, and a single step parabolic trajectory is calculated for the element. This is accomplished by analyzing independently the three components of the equation

$$\vec{x}_i = \vec{x}_{i-1} + \vec{V}_{i-1}t + \frac{1}{2}\frac{q}{m}\vec{E}\,t^2 \qquad (1)$$

for the shortest time, t, that a particle needs to reach an element face. Negative, imaginary and zero t's are, of course, rejected. A number of special conditions may occur involving round-off errors and the traversal of exceedingly small paths in the corner of elements. The treatment of these problems are discussed in detail in Section 5.62.22. Following the choice of the shortest valid time, the trajectory is advanced to another (or possibly the same) element face, where its new total energy is checked aginst its original value. The new total energy is $1/2\ m\vec{V}^2 + eV(\vec{x})$, where $V(\vec{x})$ is the bilinear value calculated for the exit location on the exit face of the element. The energy is renormalized by adjusting the magnitude of $\vec{V}$ without modifying its direction.

For volume elements that border the object, more complex E fields are anticipated so POLAR uses a slower, but more accurate, step-push method where Eq. (1) is integrated using timesteps estimated

to be approximately 0.1 of the element traversal time. At each step, $\vec{E}$ is determined by analytically differentiating the trilinear potential function (Section 4.20). The step-push method and the routines that affect it are discussed in greater detail in Section 5.62.22.

POLAR's sliced grid system (Section 4.11) forces additional computational considerations on the trajectory tracking because only a small set of potentials are stored in core at any one time. Potentials are paged in and out in slices at nodes of a constant z value. As a result, trajectory tracking is controlled by a "pusher" that sweeps back and forth in z, advancing all trajectories through the space between z and z+1. Trajectories moving opposite the pusher are written out to disk and picked up on the return pass (Section 5.62.22). Although this complicates the coding somewhat, there is little loss in efficiency, and a bonus in that trapped orbits can be simply controlled by limiting the passes of the pusher.

Magnetic field effects are also included in the POLAR ion trjectories. The techniques used to add $\vec{B}$ effects were presented in the quarterly report of December 1983 (SSS-R-84-6486). Future manual revisions will have that discussion included both here and in Chapter 5.

4.42.5 SHEATH ION DENSITIES

Once a sheath edge surface has been located and subdivided (Section 4.42.1), the input current, J_i, calculated (Section 4.42.2) and that current assigned to a representative particle, g, (Section 4.42.3), the particle trajectory is followed inward as described in Section 4.42.4. Ion densities, n_i, are determined in each cell by observing that if each trajectory, j, represents a constant current $J_{ij} = dq_{ij}/dt$, each trajectory makes a contribution to the overall element density of

$$\Delta n_{ij} = \frac{J_{ij}\,\Delta t}{\text{element volume}}$$

where Δt is the time required to cross the volume element. The total density is just

$$n_i = \sum_j \Delta n_{ij} \quad .$$

This has been dubbed the method of weighted deposition (Ref. 3-5). It can be seen that acceleration effects and convergence effects are accounted for by the Δt, and the Σ_j respectively.

This method demands a large number of particles for good statistics and we have found that the three to six particles per sheath volume element, chosen by the sheath edge algorithm (Section 4.42.1), work quite well. Problems in accuracy can still be anticipated when there exists repulsive regions within a sheath, or when the method is being incorrectly applied to an orbit-limited problem where particles might numerically diffuse into allowed trapped orbits. In this case repeated orbits would give erroneously high densities. Even in strongly space charge-limited sheaths, unusual geometry could lead to trapped orbits, so POLAR sets a user controlled limit on the number of front-to-back pushing sweeps (Section 4.42.4) to control this problem.

Finally, these sheath ion densities are known as RHOI's in POLAR and are calculated by the CURREN segment of NTERAK. The ultimate use of these densities in the Poisson solution are discussed in Section 4.43.2.

4.43 CHARGE DENSITY

4.43.1 ELECTRONS

POLAR presently considers only negative potentials. Thus electrons are repelled and are approximated by an isotropic Boltzmann equilibrium distribution, i.e.,

$$n_e = n_0 \exp(eV(\vec{x})/kT) \ .$$

This approximation may be invalid near weakly repelling surfaces, space potential barriers, and in magnetically insulated regions. Other conditions may arise where the electron distribution will not be isotropic and a Boltzmann distribution would not be justified. As positive potentials are introduced, or as other conditions require, a more elaborate electron model will be adopted.

4.43.2 *ION CHARGE DENSITY*

Ions are considered to be the attracted specie in POLAR. This, plus an allowance for possibly high Mach numbers, means that ion densities may not be determined by any local approximation. Thus ion densities are currently determined by two methods: Weak field ions (Section 4.41) known in the POLAR coding as GI's (geometric ions); and sheath ion densities (Section 4.42) known in the coding as RHOI's.

GI's are determined at the onset of a calculation and remain unchanged thereafter. The RHOI's are calculated whenever a CURREN step (the sheath ion tracking process) is called for. Immediately following a CURREN step, POLAR creates from these two data sets, an ultimate ion density list, the DION's that are used in the Poisson calculation. This is done by choosing RHOI's for elements inside the sheath, and GI's for points outside. For elements containing a portion of the sheath surface, the GI value is currently used instead of the RHOI.

4.44 THE CHARGE STABILIZED POISSON ITERATION

The Poisson equation can be written dimensionlessly as

$$-\nabla^2\phi = L^{-2}\,(n_i - n_e) \tag{1}$$

where

$$\phi = eV/kT,\ L^2 = \epsilon_o\ kT/N_o\ e^2h^2 = \lambda^2/h^2$$

is the dimensionless Debye length, N_o is the ambient density, $n_i = N_i/N_o$, $n_e = N_e/N_o$, and the Laplacian is also normalized by h^2. The calculation of n_i and n_e is discussed in Section 4.43. POLAR solves this equation on its discrete mesh of uniform spacing h, using the finite element method described in Section 4.21.

The traditional approach to the solution of equation (1) has been an explicit iteration of the form

$$-\nabla^2\phi^\nu = L^{-2}[n_i(\phi^{\nu-1}) - n_e(\phi^{\nu-1})] \tag{2}$$

where ν is the iteration index, and the charge density is determined using the potentials of the previous iteration. This method can be shown to be unstable (ref. 4-2) when the Debye length, λ, becomes small with respect to other scale lengths of the problem. This can be understood by considering that a smooth potential variation over a distance of, say, 1000 λ, would require a smooth $\nabla^2\phi$ (the 'second derivative') which is in turn given everywhere by the charge density. But, maintaining a smooth charge density distribution is difficult when any errors in determining $(n_e - n_i)$ are multiplied by a huge L^{-2}. There is one effective remedy to this dilemma due to Parker (ref. 4-2), but the process reported here appears to be more efficient in the short Debye length limit. This method involves the combination of two concepts. One uses a partial implicitization of the repelled density (n_e, here) (ref. 4-3). The other simply reduces the charge density to an acceptable level whenever the first method is inadequate.

Suppose a plasma of ambient density N_o and temperature T consists of Boltzmann electrons, $N_e(\vec{r}) = N_o \exp(\phi(\vec{r}))$ and ions of known density $N_i(\vec{r}) = N_o\, n_i(\vec{r})$. The normalized charge density is then given by

$$q(\vec{r},\phi^{\nu}(\vec{r})) = L^{-2}\,[n_i(\vec{r}) - \exp\,(\phi^{\nu}(\vec{r}))] \tag{3}$$

Equation (3) may be linearized about the previous potential iterate

$$q(\phi^{\nu}) = q(\phi^{\nu-1}) + q'(\phi^{\nu-1}) * (\phi^{\nu} - \phi^{\nu-1})$$

where $q' = \partial q/\partial\phi$, and the $\vec{r}$ dependence has been dropped for clarity. With this expression we may write the implicit Poisson iteration scheme

$$-\nabla^2\phi^{\nu} - q'(\phi^{\nu-1}) * \phi^{\nu} = q(\phi^{\nu-1}) - q'(\phi^{\nu-1}) * \phi^{\nu-1} \tag{4}$$

Though it is not immediately obvious, the implicit character of (4) makes it more stable than scheme (2). This can be understood by realizing that in equation (3) the electron density was treated as an independent variable, whereas in (4) the electron density is determined simultaneously with the potential, both being consistent with the ion density.

The finite element approximation (Section 4.21) to (4) produces the matrix equation

$$\sum_e \left(\underset{\sim}{W}^{(e)} - \overline{S'}^{(e)}\,\underset{\sim}{V}^{(e)}\right) * \underset{\sim}{\phi}^{\nu} = \underset{\sim}{S} - \overline{S^r}^{(e)} * \underset{\sim}{\phi}^{\nu-1} \tag{5}$$

where S is derived from q by the following analysis:

For $L \gtrsim 1$, S is simply the total charge associated with each node, q. However, for $L \ll 1$, numerical noise and features like a sheath edge which may span only a few λ, become incorrectly amplified

when the q determined at a point becomes multiplied by the entire nodal volume. When it is not possible to reduce the zone size, stability can be preserved by replacing Q (and Q') with a reduced value S (S') which is calculated to be the maximum allowable charge for the element. Because of the artificial amplification argument, S is often the more realistic total for an element. Before deriving S, we define the barometric potential $\phi_b = \ell n(n_i)$ which is the potential for which $Q = 0$ and note that it is important that $S \to Q$ as $\phi \to \phi_b$ if quasineutral regions are to be modeled correctly. To determine S, consider a capacitor with potential difference $(\phi_b - \phi)$, area h^2, and a separation of h. The charge q_c on this capacitor is given by

$$q_c = C\Delta V = \frac{\epsilon_o h^2}{he} (\phi_b - \phi)kT$$

In the units of our previous q, q_c becomes

$$q_M = \alpha(\phi_b - \phi) = \alpha(\phi_b - \phi)$$

which is the maximum allowable charge per element, with the parameter α, adjusted to insure that q_M is maximized. Thus at each node, we choose for the charge

$$|S| = \min(|q_M|, |q|)$$

with

$$S' = \begin{cases} -\alpha & \text{for } S = q_M \\ L^{-2} & \exp\phi \text{ for } S = q \end{cases}$$

Actually, S and S' are smoothed in POLAR to decrease numerical noise and spatial potential oscillations. This involves forming the linear screening term

$$SCRN = \frac{1}{3}\,(2 \cdot S'(\bar{\phi}) + \bar{S}')$$

where

$$\bar{S}' = \sum_I V(I) \cdot S(\phi(I))$$

where I indexes the nodes of an element and V(I) is a nodal volume normalized by a cubic element volume of one; similarly for ϕ. Also smoothed is the nodal charge Q(I),

$$Q(I) = \frac{1}{3}\,(2 \cdot S(\bar{\phi}) + S(\phi(I)))\ .$$

Additionally, POLAR can output the value $S(\bar{\phi})$ as QUSD, the element centered charge actually chosen by this algorithm.

Finally, the Q(I), and SCRN are used for S and $\overline{S'}^{(e)}$ in Eq. (5).

The effect of this algorithm is this: If a problem has been specified where a boundary potential would be screened in less than a zone or two (the limit of any code's resolution), sufficient sheath charge will be redistributed so as to allow the potential to be screened over the minimum number of zones that are consistent with stability.

The charge stabilization algorithm is effected by the subroutine, QSELT, and QSCRN which are further described in Section 5.50. Tests of this method are presented in Appendix C.

4.50 CHARGING MODEL

4.51. CIRCUIT MODEL

POLAR's circuit model was introduced in Section 3.40. Here we describe the details of the time-integration of the basic charging equation, 3.40-1, repeated here.

$$\underset{\sim}{I}(t) = \underset{\approx}{C}\,\frac{d}{dt}\,\underset{\sim}{V}(t) - \underset{\approx}{\sigma}\,\underset{\sim}{V}(t)$$

where I is current, C capacitance, σ conductance, and V is surface voltage. We first difference this equation, evaluating the conductance term at the advanced time (implicit) and the current at the retarded (explicit) time.

$$\underset{\approx}{C}\,\frac{(\underset{\sim}{V}(t_2) - \underset{\sim}{V}(t_1))}{t_2 - t_1} - \sigma\,\underset{\sim}{V}(t_2) = \underset{\sim}{I}(t_1)$$

or

$$\left[\frac{\underset{\approx}{C}}{t_2 - t_1} + \underset{\approx}{\sigma}\right] * (\underset{\sim}{V}(t_2) - \underset{\sim}{V}(t_1)) = \underset{\sim}{I}(t_1) - \underset{\approx}{\sigma}\,\underset{\sim}{V}(t_1)$$

Section 3.40 explored the difficulties associated with the explicit current dependence. The implicit approach replaces $I(t_1)$ with $I(t_2)$, approximated as

$$I(V(t_2)) = I(V(t_1)) + \left.\frac{dI}{dV}\right|_t * (V(t_2) - V(t_1))$$

Substituting, we get the implicit formulation

$$\left(\frac{\underset{\sim}{C}}{\Delta t_1} + \underset{\sim}{\sigma} - \frac{\underset{\sim}{dI}}{\underset{\sim}{dV}}\right) \cdot \left(V(t_2) - V(t_1)\right) = \underset{\sim}{I}(t_2) - \underset{\sim}{\sigma}\, V(t_1) \tag{4.51-1}$$

where $\Delta t = t_2 - t_1$.

The $d\underset{\sim}{I}/d\underset{\sim}{V}$ term stabilizes the problem by increasing the matrix entries on the diagonal. Physically, the ∂I (surface=i)/∂V (surface i) will be non-positive since surfaces charge more slowly as the voltage decreases (becomes more negative). Currently, no surface to surface interactions are included so the current derivative matrix is diagonal.

Unfortunately, due to the complexity of the plasma interaction, the dI/dV in Eq. (4.51-1) is not trivial to evaluate; so we must use a predictor-corrector type approach. We may omit the dI/dV term from (4.51-1) to return to the explicit form, which may be solved for an estimate of $\underset{\sim}{V}(t_2)$. Anticipating the possible instability of the explicit form, the $\Delta\underset{\sim}{V}$ from t_1 to t_2 is limited by the input quantity DVLIM. With this new, albeit rough, estimate of $\underset{\sim}{V}(t_2)$ we could run a complete Poisson current sequence to get a ΔI estimate. But, because of the expense of a complete Poisson current step, we leave the ion currents constant (otherwise a particle pushing step would be required), and obtain ΔI from the electron current algorithms alone (4.50). This will provide an estimate of $\Delta I/\Delta V$ sufficient to solve (4.51-1) implicitly for $V(t_2)$.

To discuss the solution to Eq. (4.51-1), we abbreviate it as

$$\underset{\sim}{M} \cdot \Delta\underset{\sim}{V} = \underset{\sim}{R} \tag{4.51-2}$$

where M include the capacitance, conductance and dI/dV matrices. When

the number of vector components is not too great ($\lesssim$1000), we have found the Incomplete Cholesky Conjugate Gradient (ICCG) method to be an efficient means of solving (4.51-2) (see Section 4.30).

Experience has shown that ICCG is most effective when M is diagonal or nearly diagonal (diagonal elements >> off diagonal elements). To see how $\underset{\approx}{M}$ may be improved, write (4.51-2) as

$$(\underset{\sim}{M}_1, \underset{\sim}{M}_2, \text{---} \underset{\sim}{M}_C)\Delta\underset{\sim}{V} = \underset{\sim}{R} \qquad (4.51\text{-}3)$$

If surface 1 connects to infinity (conductor 0) and conductor C, and we consider only the capacitance, we would have

$$\underset{\sim}{M}_1 = \begin{pmatrix} -C_{10} - C_{1C} \\ - \\ - \\ C_{1C} \end{pmatrix} \quad \text{and} \quad M_C = \begin{pmatrix} C_{1C} \\ - \\ - \\ -C_{C0} \end{pmatrix},$$

where $C_{iC} >> C_{10}, C_{C0}$. (That this is a legitimate example, see the complete sample problem worked in Appendix A.) It is the off diagonal entries of C_{1C} that need to be removed. This is accomplished by transforming (4.51-3) to

$$(\underset{\sim}{M}_1, \underset{\sim}{M}_2, \text{-}, \underset{\sim}{M}_C + \underset{\sim}{M}_1) \begin{pmatrix} \Delta V_1 - \Delta V_C \\ - \\ - \\ - \\ \Delta V_C \end{pmatrix} = \underset{\sim}{R}$$

A column transformation for each surface will "clean up" the upper right triangle of $\underset{\approx}{M}$, while producing a transformed potential vector where surface potentials have been replaced by the potential difference between the surface and the underlying conductor. The lower left triangle of $\underset{\approx}{M}$ is diagonalized and symmetrized to the upper

by similar row additions with corresponding transforms of $\underset{\sim}{R}$ on the right hand side. Currents (in $\underset{\sim}{R}$) to surfaces remain unchanged, but currents to conductors are replaced as

$$I_C \rightarrow I_C - \sum_S I_S$$

where the sum is over all surfaces connected by capacitance or conductance to the main conductor, C; other conductors are treated as surfaces and referenced to C_1. The $\underset{\approx}{\sigma}$ and $\underset{\sim}{V}$ on the right hand side are treated identically to $\underset{\approx}{\sigma}$, $\underset{\approx}{C}$ and $\underset{\sim}{V}$ on the left. Finally, the dI/dV matrix is calculated for the transformed I and V guaranteeing that it will be diagonal.

Another feature of POLAR's charging model will change the previous transformations. This happens when NTERAK recognizes that the lowest estimate of the ion flux (ambient ram conditions including shadowing) exceeds the greatest estimate of the primary energetic electron flux, and the surface potential is near zero. Physically, if kthe magnitude of the surface potential drops to near $-k\ T_{e,cool}$ (typically 1 volt) the flux of cool electrons will prevent the surface from charging positive. These millivolt sensitivities are difficult to follow in a code that is concerned with kilovolt charging. Thus, such a surface will be spotted (this is currently implemented) and have its potential fixed at $-k\ T_{e,cool}$ for the upcoming charge cycle.

This will also occur when a presently nonexistent routine in NTERAK will recognize the required conditions for a surface's potential to be controlled by a $\vec{E} \cdot \hat{n} = 0$ condition on the surface. This may occur when a secondary or photoelectron space charge density becomes large enough to form a small barrier to the low energy portion of the emitted spectrum. Such a surface will be flagged and get its potential floated in the Poisson solution according to the $\vec{E} \cdot \hat{n} = 0$ condition, while being held fixed during the circuit solution.

This fixing is not compatable with the $\underset{\sim}{V}$ transformation, so POLAR is forced to solve the circuit model with the original equations. Accuracy does not suffer, but ICCG will require more time for the same level of convergence.

Other conditions will also arise that will require a surface cell to be held at a fixed potential.

4.52 ELECTRON CURRENTS, PRIMARY, SECONDARY

The POLAR electron environment is described in Sections 3.15 and 3.31. This section explains the integration procedures used to obtain the net electron currents to a surface due to primary, secondary, and backscatter electrons. Further information concerning the code mechanics and subroutine relations can be found in Section 5.70.

4.52.1 SECONDARY ELECTRONS

The secondary electron yield coefficients used in this section (4.52.3, 4.52.4, 4.52.5) are calculated using the proven techniques and theories developed for NASCAP. The reader is currently referred to Reference 4-1 for a description of these methods.

4.52.2 BACKSCATTER ELECTRONS

See Section 4.52.1.

4.52.3 INTEGRAL OF THE MAXWELLIAN DISTRIBUTION

The current J_M of electrons of charge q to a surface at voltage V due to the Maxwellian portion of the spectrum is given by

$$J_M = \frac{qF}{(kT)^2\pi} \int_0^{2\pi} d\phi \int_0^{\pi/2} \sin\theta \cos\theta \, d\theta \, g(E, \psi(\phi,\theta))$$

$$* \int_L^{\infty} K \, dK \exp[-(qV + K)/kT]$$

$$L = \max(-qV, 0)$$

The lower kinetic energy limit L is chosen to exclude orbits that cannot energetically connect to infinity. The function g is a function of the pitch angle ψ (see Section 3.31) and total particle energy E. We are concerned here with the integral over kinetic energies, so the angle integrals will be performed assuming g to be a constant function of value unity, so as to give the correct current for an isotropic flux.

The flux of electron generated secondary and backscatter electrons is obtained by adding a yield function $Y(K,\phi,\theta)$ to the above integrals (Section 3.33). POLAR presently replaces Y with an angle averaged $\bar{Y}(K)$ for the isotropic case. Thus

$$J_{MS} = \frac{qF}{(kT)^2} e^{-qV/kT} \int_L^{\infty} K \, e^{-K/kT} \cdot \bar{Y}(K) \cdot dK$$

where the primary current J, is obtained by setting $\bar{Y} = 1$.

For arbitrary $\bar{Y}$, the integral must be performed numerically. It is also desirable to divide up the spectrum in a manner that covers the larger fluxes at low energies without ignoring the high energy tail of the Maxwellian spectrum. A logarithmic spacing is accomplished by the substitution

$$K = -kT \ln x \ .$$

Thus we have

$$J_{MS} = qF \int_{x1}^{xu} \bar{Y}(K(x)) \ln x \, dx$$

where $xu = e^{-L/kT}$, and $x1 = 0$. Since $\ln x$ is singular at $x = 0$, the lower limit is set to $x1 = 0.01 * xu$ and the omitted portion of the spectrum approximated by $x1 * \ln(x)$. The summation is performed using Simpson's rule and 20 points.

4.52.4 INTEGRAL OF THE POWER LAW DISTRIBUTION

The discussions of angular dependence in the flux integral given in Section 3.31 and 4.52.3 also apply here, so we will limit this treatment to the energy integral

$$(A) \quad J = aq \int_{KL}^{KU} \bar{Y}(k) \cdot (K + qV)^{-(\alpha+1)} \, K \, dK$$

where J is current, $\bar{Y}$ is secondary or backscatter yield, K is kinetic energy, a is a constant ($a = \pi A$ for isotropy), q is charge, and V is surface potential. For the limits we choose

$$KL = MAX\ (0,\ EL - qV)$$
$$KU = MAX\ (EU,\ EU - qV)$$

where EL is a physical cutoff (default = 100 eV) and EU is imposed *sufficiently high as to avoid significant error* (default = 1×10^9 eV).

The first step is the transformation

$$X = K + qV \ ,$$

which gives

$$J = aq \int_{XL}^{XU} \bar{Y}(K(X)) \cdot [X^{-\alpha} - qV\ X^{-(\alpha+1)}] \, dX$$

where XL = max(qV, EL) and XU = max(EU + qV, EU).

Since α can be as large as 3.0, this spectrum is strongly peaked towards lower energies, which implies that a nonuniform spacing of integration points is desirable. This is easily accomplished by the substitutions,

$x = y^{-1/(\alpha-1)}$, for the first term, and

$x = z^{-1/\alpha}$, for the second term

of the previous integral, which leads directly to

$$J = aq\left[\frac{1}{\alpha-1}\int_{yl}^{yu} \bar{Y}\, dy + \frac{1}{\alpha}\int_{Zl}^{zu} \bar{Y}\, dz\right] \quad .$$

where $yl = xl^{1-\alpha}$, etc. Notice that these choices produce integration weights of value unity. The numerical integration is done over 20 points spaced evenly in y and z. These transformations are very strongly biased towards the lower energies thus the upper cutoff was chosen quite high (1×10^9 eV) to force good coverage of intermediate energies where $\bar{Y}$ might be peaked.

A method with variable bias was also investigated. This utilized for Eq. (A), the substitution

$$K = x^{-1/\beta} - S$$

$$dK = -\frac{1}{\beta} x^{-(1+1/\beta)}\, dx = w(x)\, dx$$

This method has the ability to adjust the bias with β, and center the integration points about an energy related to S. This method is inherently slower due to the calculation of the weights w(x), and appeared to offer no great advantage over the previous method. It is not presently used by POLAR, but remains an option.

4.52.5 INTEGRATION OF GAUSSIAN DISTRIBUTION ELECTRONS

This treatment is limited to the energy integral of the Gaussian electron distribution. The angular dependence has been integrated assuming isotropy as discussed in Section 3.31 and 4.52.3. From Section 3.31, the integral of interest is

$$(A) \quad J_S = \pi B \int_0^\infty \bar{Y}(K) \exp[-(K-K_o)^2/\delta^2]k \, dK \quad .$$

As in the previous Sections 4.52.1 and 4.52.2, the inclusion of the angle averaged yield function $\bar{Y}(K)$ will make J the current of backscatter or secondary electrons (see Section 3.33); with the omission of $\bar{Y}$, J becomes the incoming primary flux.

This integral is performed numerically by an 8 point Hermit integration scheme (Ref: Handbook of Mathematical Functions, U. S. Dept. of Commerce, National Bureau of Standards, Applied Math Series No. 55, pp. 924, 1964).

The weights and absissa are:

i	x_i	$w(x_i)$
1,5	$\pm$0.38119	6.61147×10^{-1}
2,6	$\pm$1.15719	2.07802×10^{-1}
3,7	$\pm$1.98166	1.7078×10^{-2}
4,8	$\pm$2.93064	1.99604×10^{-4}

Equation (4.52.3-A) can thus be approximated by

$$(B) \quad J_S = \pi B \sum_{i=1}^{8} w(x_i) \cdot \bar{Y}(K_i) \cdot \exp[-\Delta K_i^2/\delta^2] \cdot K_i \cdot \Theta(K_i - KL)$$

where

$$K_i = K_0 + X_i * \delta$$

$$\Delta K_i = X_i * \delta$$

and

$$\Theta(K_i - K) \begin{cases} = 1 \text{ for } K_i - KL > 0 \\ = 0 \text{ for } K_i - KL < 0. \end{cases}$$

KL is chosen as max(0, -qV) which excludes energetically trapped orbits.

J_S as given by B has the undesirable property of being discontinuous with respect to K due to the Θ function. This feature can be removed by calculating J with the same scheme (omitting $\bar{Y}$). J will be discontinuous at exactly the same values of K as J_S. We can then form a smooth J_S^* as

$$J_S^* = \frac{J_S}{J} \cdot J_a$$

where J_a is the analytic solution to the primary flux integral derived below:

$$J_a = qA \int_L^{\infty} \exp\left[-\left(\frac{K - K_0}{\delta}\right)^2\right] K \, dK$$

set

$$X = (K - K_o)/\delta$$

so

$$J_a = \pi B \int_Z^\infty \exp(-X^2)\ (X\delta + K_o)\ dX$$

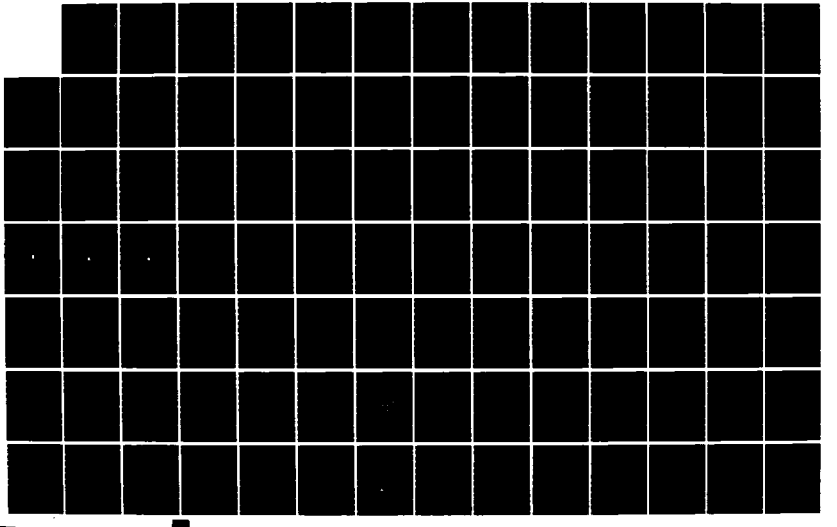
where

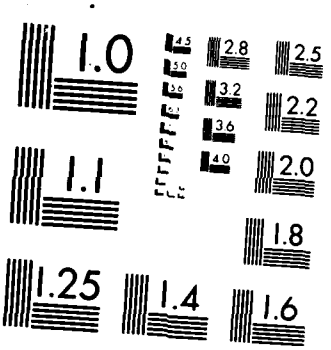
$$Z = \frac{KL - K_o}{\delta} .$$

Integrating

$$J_a = \frac{q\pi B\delta^2}{2} \left[\exp(-Z^2) + \frac{\sqrt{\pi}}{\delta} \left((K_o + |K_o|\ \mathrm{erf}(|Z|)\right)\right]$$

AD-A173 758 POLAR USER'S MANUAL(U) S-CUBED LA JOLLA CA 2/3
J R LILLEY ET AL. OCT 85 SSS-R-86-7563 AFGL-TR-85-0246
F19628-82-C-0081
UNCLASSIFIED F/G 22/1 NL





4.53 ION SURFACE CURRENTS

For the attracted ion currents, POLAR tracks particles from a sheath surface to an object surface. Experience has shown that if surface currents are simply accumulated from incident particles, the currents are noisy and lead to non-physical charging behavior. Numerous reasons exist for this noise: tracking errors, potential field irregularities, too few trajectories, etc. While these problems have all been studied, it remains desirable to have a smoothing algorithm for the ion surface currents.

The ion currents to be smoothed are derived from the "dead-list" of particles produced from the particle pushing module CURREN (4.40). This dead-list contains the particle's weight, final position, final velocity, and initial energy and the surface the particle landed on. To speed up the surface current calculation and to reduce noise, the dead-list is condensed to the SRFC list, ordered by surface number. The particle weights are added since they represent the ion current. The particle weights are also used to weight the averages of spatial and energy information. An average position on the cell, angle of incidence, and particle energy are calculated as follows:

$$\bar{X}_i = \frac{\sum_{k=1}^{N} w_k X_{ik}}{\sum_{k=1}^{N} w_k}$$

where w_k is the weight (current contribution) of particle k, N is the number of particles striking the surface, and X_i is the i^{th} component of the position vector. The average velocity and energy are found similarly.

The adopted smoothing algorithm is a two-step process wherein the raw surface currents are distributed to nodes, and then re-distributed back to surfaces. This simple algorithm is illustrated in Figure 4.53/1. Given an averaged particle at the point P, on surface a, its current is shared in a bilinear fashion to the vertices (nodes) of the surface, producing a node current I_i.

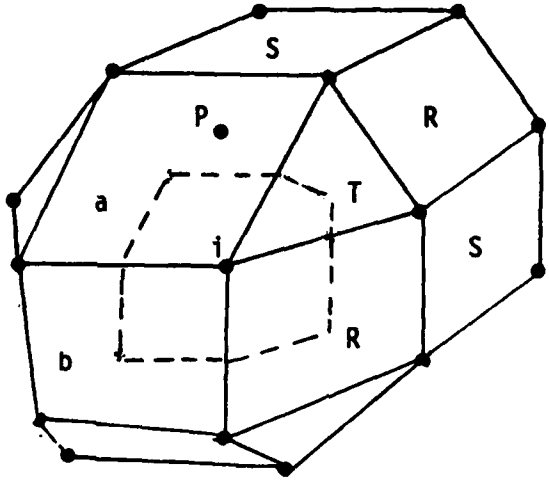


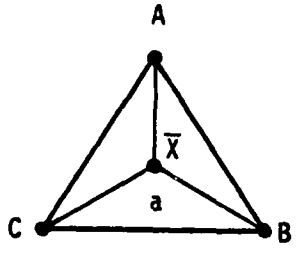
Figure 4.53/1.

Since two different types of surface occur, triangles and rectangles, two different methods are used. For triangles, the bilinear weight of a corner is area of the triangle opposite the corner over the total surface area (see Figure 4.53/2).

$$wb_{sa} = \frac{\text{area a}}{\text{area of triangle ABC}}$$

or

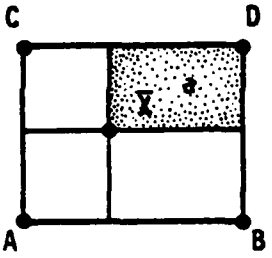
$$wb_{sa} = \frac{\left|\vec{\geq}\ (\vec{X}-\vec{C}) \times (\vec{B}-\vec{C})\right|}{\left|(\vec{A}-\vec{C}) \times (\vec{B}-\vec{C})\right|}$$



where w_{sn} is the bilinear weight, $\bar{X}$ is the particle position, and s and a refer to surface and node.

Figure 4.53/2. Bilinear weighting of triangular surfaces

For rectangular surfaces, the particle position divides the rectangle into four smaller rectangles. Then the weight is found by dividing the area of the opposite small rectangle by the area of the surface (see Figure 4.53/3).



$$wb_{sa} = \frac{\text{area of a}}{\text{area of ABCD}}$$

Figure 4.53/3. Bilinear weight (wb_{sa}) of a rectangular surface.

Thus we have for Figure 4.53/1

$$\Delta I_i = wb_{ai} \cdot SRFC(a)$$

where the Δ indicates that this is a current increment. The complete I_i is never formed.

The next step is to share the node current back to the surfaces. We may derive a simple sharing formula by forming a nodal current density, ΔJ_i, as

$$\Delta J_i = \Delta I_i / A_i$$

where A_i is the area associated with node i;

$$A_i = \sum_{j=1}^{m_i} A_j / n_j$$

where the A_j are areas of the m surface cells adjoining node i, and n_j is the number of vertices of each adjoining surface cell. A_i is bounded by the dashed line in Figure 4.53/1.

The ΔJ_i is redistributed to surfaces adjoining node i in proportion to the area each contributed to the node area. Thus, surface b would receive a final surface current increment ΔI_b,

$$\Delta I_b = \frac{A_b}{n_b} \Delta J_i$$

Finally, we may combine the three previous equations into a node to surface weight factor ws_{ij}. Thus the final surface current, ΔI_b, is obtained from the intermediate node current ΔI_i as

$$\Delta I_b = ws_{ib} \Delta I_b$$

$$ws_{ib} = \frac{A_b n_b}{\sum_{j=1}^{m_i} A_j / n_j}$$

This process is performed for each surface adjoining the i nodes of surface a. The final smoothed surface currents are accumulated as this overall process is repeated for all the surfaces of the object.

An important feature of this algorithm is that a uniform flux to an irregular object will produce surface currents exactly proportional to the surface areas as would be expected. To see this, consider the quasisphere of Figure 4.53/1. This object has three types of surfaces, with areas S, R, and T; and only one type of node. A uniform flux of particles, if tracked accurately, will produce uniform node currents. Summing $w_{ij} * I_i$ around the vertices of a surface, we see that the resulting surface current will be proportional to the surface area, and inversely proportional to the factor in the denominator of the expression for ws_{ij}, which is constant for our example. The ws_{ij} are properly normalized. This can be easily seen in our example by summing the ws_{ij} around a node;

$$\sum_{k=1}^{n_j} \frac{A_k/n_k}{\sum_{j=1}^{n_j} A_j/n_j} = 1.0$$

since the indices k and j range over the same surfaces.

At our present stage of development, we are simply dividing and redistributing the node currents according to the relative areas of adjacent cell, but provisions have been made for a more comprehensive treatment. For example, spatial and electrical information could be used to avoid non-physical sharing of surface currents; such as redistributing ion currents to a surface with a large positive potential, or around corners.

REFERENCES, CHAPTER 4

4-1 "NASCAP Programmers' Reference Manual," S-CUBED Report SSS-R-82-5443 (DRAFT), March 1982.

4-2 Parker, L. W. and E. C. Sullivan, NASA Report No. NASA TN D-7409, 1974.

4-3 Parker, L. W., "Calculation of Sheath and Wake Structure About a Pillbox-Shaped Spacecraft in a Flowing Plasma," Proceedings of the Spacecraft Charging Technology Conference, AFGL-TR-77-0051, NASA TMX-73537, 1977.

4-4 Laframboise, J. G. and L. W. Parker, "Probe Design for Oribit-Limited Current Collection," Phys. of Fluids, Vol. 16, N5, 1973.

5. POLAR CODE STRUCTURE

This chapter is designed to provide insight into the internal workings of POLAR. Whenever possible, actual subroutine names and variable names will be used.

5.10 TOP DOWN VIEW

POLAR is actually five s' 'alone programs that communicate through a minimum number of fil This approach allows a high degree of flexibility in model building while minimizing the amount of unnecessary computing. These programs are VEHICL, ORIENT, NTERAK, SHONTL, and PLOTTR. Their functions are described below. Whenever scratch files are used, they are assigned and disposed of automatically. In general, only two files are needed to allow communication between the four modules.

VEHICL is the object definition program. It utilizes much of the user-oriented object definition procedures developed at S-CUBED for NASCAP. With VEHICL, one uses a variety of basic building blocks to define the vehicle to be modeled on a variable sized 3-D grid. One also defines all of the surface properties and underlying conductors. VEHICL then completes the vehicle electrical model and creates a number of "connectivity" tables to accelerate NTERAK execution. This information is written on two files, 11. and 19., which carry this information to the other modules.

ORIENT is the attitude control program. User input is simply the dominant plasma flow direction viewed from the vehicle. If necessary, ORIENT will rotate the vehicle and object grid so as to keep the wake direction predominantly in the +Z direction. ORIENT will also restructure most of file 11 so that it will have the correct sliceability. The restructured file 11. contents will either be copied back to file 11., or output to a new file 11S, where S

represents a user-supplied suffix. A set of six 11S files would catalog all of the possible coordinate orientations, and allow for all Mach vectors.

NTERAK is the biggie that actually calculates the vehicle-plasma interaction. The internal workings and I/O are the subject of most of this document, but it should be emphasized that once a Mach vector has been defined, the extended computational grid will be "burned" into the 11S file. The 11S maintains a complete restart-continue capability, but a fresh file must be used if the Mach vector is to be changed.

SHONTL is the machine independent plotting package. It is designed to be run semi-interactively. By this we mean that plotting directives are entered by a "keyword" input system that function in both batch and interactive environments, but nothing is plotted until the session is concluded and PLOTTR is executed. This somewhat cumbersome procedure is necessary to maintain the machine independence of SHONTL. SHONTL can be used after any of the previous three modules to graphically check the work progress at any given stage. SHONTL reads from file 11, and communicates with PLOTTR through file 2. PLOTTR is the machine-dependent plotread package that reads the generic pen-move and vector commands written on file 2 by SHONTL, and translates these commands to the local graphics package.

5.13 NTERAK

The program NTERAK is the heart and brain of POLAR. It contains all of the physical models and computational techniques described in the previous chapters. The purpose of this section is to outline the structure of NTERAK. Further descriptions of the functional content of the more important routines can be found in the indicated sections. For clarity, utility routines have been omitted from the following chart:

Level 1

NTERAK: Main executive routine
Calls INPUT
SPACE
PWASON

Level 2

INPUT: Controls the keyword input of options and parameters (6.40) and opens files 9 and 11 (5.30).

SPACE: Set up the computational grid (5.20).
Calls MRBUFF.

PWASON: Control of the Poisson iteration and tests for convergence (4.21.5).
Calls IONDEN
Calls CONGRD

Level 3

MRBUFF: Initializes the vector properties that effect their paging from core to disk.

IONDEN: Controls the calculation of the ion densities (3.20, 4.40).

CONGRD: Performs the conjugate gradient calculation and tests CGM convergence (4.21.1).

Calls COPROD
URSETO

loop on | COPROD
| PUTDAT
| RUPDAT
| UUPDAT

Level 4

COPROD: Called first to calculate $\underset{\sim}{\underset{\sim}{M}} \cdot \underset{\sim}{\phi}$ to estimate initial residuals, r (4.21.1) and to calculate the linear screening for the Poisson iteration (4.46).

: Loop calls calculate $\underset{\sim}{\underset{\sim}{M}} \cdot \underset{\sim}{U}$

Calls BUFCLR
BUFSET GRID
PAGER XYGRID
CURCEL ZBNDRY
VERTIO YBNDRY
ELEMNT XBNDRY
DCVCEL GETSCR
FORFIL PCUBES
BACFIL ECUBES

URSETO: Calculate r . r and initialize u (4.21.1).

PUPDAT: P equation.

RUPDAT: r equation.

UUPDAT: u equation.

Those level 5 routines, called by COPROD that are important to understand are discussed in 5.20 on the displaced grid system.

5.20 SLICE GRID SYSTEM

The displaced slice grid system is designed to provide the computational space in which to solve Poisson's equation for the shuttle orbiter, including a wake extending many spacecraft lengths. Hence the grid must continue for an arbitrary length in the plasma flow direction.

To facilitate this, the grid is composed of a variable number of XY slices, stacked along the Z axis, rather like a loaf of sliced bread. Figure 5.20/1 illustrates a displaced grid system along with a number of important parameters. Objects are defined on a rectangular NXOB x NYOB x NZOB grid (5.11, 6.20). In all cases, the object grid will have been rotated by ORIENT such that the dominant component of the plasma flow vector VMACH is oriented along the +Z direction. As shown in the figure, the grid follows VMACH by stepping ± 1 unit in the X and Y direction every IDELX and IDELY mesh units along the Z direction (the ± 1 step follows the sign of IDELX or IDELY). These step intervals are calculated in the routine N.SPACE according to the relation (FORTRAN)

$$IDELX_{(y)} = VMACH(Z)/VMACH(X)_{(y)} \pm 0.5$$

where the $\pm$ follows the sign of VMACH(X). Since IDELX is an integer, the 0.5 centers the velocity ratio between integral increments.

The computational grid must enclose the object grid with a minimal amount of wasted space. To accomplish this, the routine SPACE references the two grids at the point shown in Figure 5.20/1, then calculates the NXGRTH(NYGRTH) necessary to fit the grids together. When the user anticipates the need for additional work space, the INPUT parameters NXADON and NYADON can increase the X-Y grids without

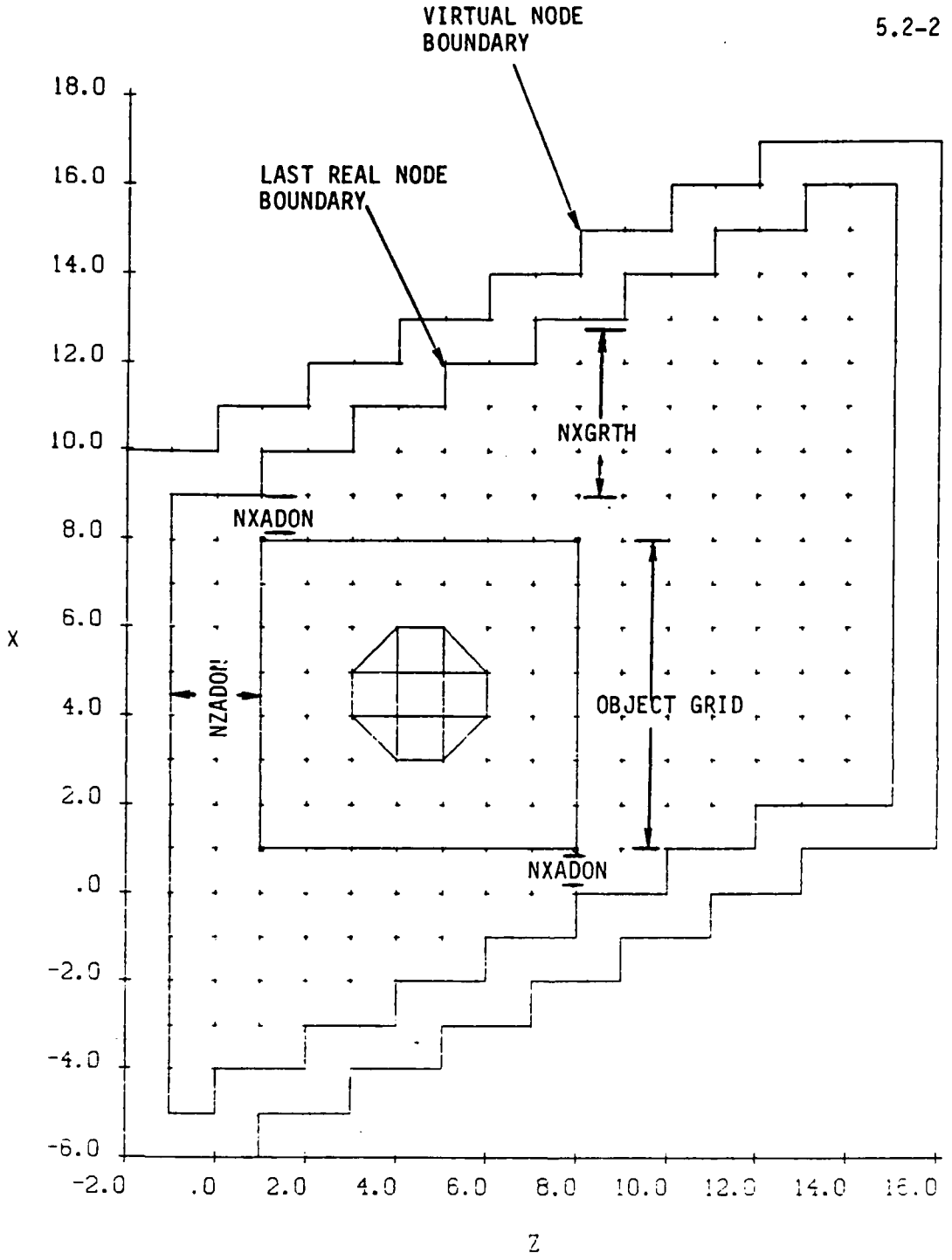


Figure 5.20/1. A X-Z cut of a typical NTERAK computational mesh showing the object definition grid and the computed (NXGRTH, IDELX) and user-specified (NXADON, NZADON, NZTAIL) grid parameters.

requiring a redefinition of the object space. The GRTH and ADON parameters are included into the final NX x NY dimensions.

The computational space is characterized by the following parameters:

NXOB,NYOB,NZOB	=	The real node dimension of the object grid along the X, Y, and Z directions.
NX	=	NXOB + NXGRTH + 2 x NXADON
	=	the real node dimension of a slice along the X direction.
NY	=	NYOB + NYGRTH + 2 x NYADON
NZ	=	NZOB+NZTAIL + NZADON

The NZADON and NZTAIL are also inputs to NTERAK and are currently limited to total a Z node count of 100. This limit could be extended indefinitely subject to available disk storage and budget limitations. This last feature is the intended result of NTERAK's data management system. This system allows models to be constructed primarily on disk with the computer "core" used to perform arithmetic on small volumes of space.

5.21 SLICE MACHINERY

The grid machinery consists of routines designed to move or page grid information between disk and core. In both media, information for each of the vectors ('p', 'r', 'u', etc. (4.21.1) are called vectors even though they are scalars at a particular grid point) involved in a calculation is organized into individual one-dimensional arrays corresponding to each X-Y slice.

NTERAK distinguishes between two types of nodes, real and virtual. At real nodes a problem's variables are truely variables. Virtual nodes exist as the outer boundary of the problem, where values are generated according to the boundary conditions. There may be one or two virtual nodes beyond the edge of a slice (see Figure 5.20/1) depending on the proximity of a step. Only real nodes are paged in and out. Each vector slice has a real node length NX x NY with the assumed convention that the X coordinate varies the fastest.

In order to relate the slices to one another and perform arithmetic with minimal confusion, it is necessary to adopt a standard coordinate system. There are two possible choices; object and slice coordinates. We have chosen object coordinates as the primary system, but slice coordinates are sometimes required by lower level routines. Object coordinates reference the "least" real node of the object grid as (1,1,1) (see Figure 5.20/1). By "least" we mean the node for which NX x NY x (Z-1) + NX x (Y-1) + X is a minimum. Thus the least real node object coordinates of some slices may be less than zero, while the slice coordinates of the least node will be (1,1,ZSLICE) where ZSLICE numbers from 1 to NZ. The relationship of the slices to each other is written in the common block MESHY by the routine GRID. This block contains the object coordinates of the least real node of each slice. For further detail concerning subroutines and common blocks see Appendices A and B.

NTERAK's grid machinery exchanges information between disk and a special common block called CBUF. The core location of CBUF is such that its addresses are the highest of all other data and instructions. On many machines this allows the CBUF length to be extended or reduced dynamically during execution to precisely match storage requirements. The allocation and addressing of data space in CBUF is controlled by three routines (see Section 5.30), MRBUFF, BUFCLR, and BUFSET. MRBUFF is called only once from the level 2 routine SPACE and initializes the array VPROPS which contains all of the individual vector properties that affect the vector's handling storage requirement in CBUF and on disk.

Subsequently, any computational process requiring vector slice transfers to/from disk will first call BUFCLR. BUFCLR clears the ADDRES common block which contains the CBUF addresses, releases any dynamical requested core, and resets other storage control variables.

The next step is to call the routine BUFSET, passing it a list of up to four vector names and the number of slices that will be needed in core. BUFSET will assign each vector a region in CBUF, and list this addresse in the common ADDRES. BUFSET may be called repeatedly for a total of 12 vectors. For example, in an operation requiring three slices of 'R', 'R' would be assigned the address sequence; A, B, C, A, B, C, A, B, ... , for all 1-NZ slices. The actual transfers are effected by PAGER, which can read, write, or read-write slices. Suppose that 2, 3 and 4 were in core starting at the CBUF address B, C and A, respectively. If PAGER were called to read-write 'R', for slices 3-5, 'R' slice 2 at B would be written on disk, and slice 5 would be read into CBUF at B. The R slices 3 and 4 would remain untouched.

Some other routines that are commonly used with this grid machinery should be mentioned. One is XYGRID, which looks in the MESHY common block and returns the X and Y limits of a slice in object coordinates. The others are XBNDRY, YBNDRY, and ZBNDRY. These routines will take an element index (equal to the least index of its eight nodes) and determine which of the element's eight nodes are real or virtual (boundary nodes). This information is kept in the array MBXYZ(8). To form an element on the staggering grid, VERTIO is called to pick four nodes from each of the two bordering slices totaling eight vertices. Individual words of a vector at a node are located in CBUF by the statement function CBUFAD (Appendix B).

In retrieving these words, the MBXYZ array is consulted to determine if a node is virtual. If so, the boundary value (currently zero) will be used for that node.

The routine GETSCR is a cousin to VERTIO. It will 'GET' or 'REP' the element centered quantities 'GI' or 'SCRN'. 'GI' is the name given to the normalized ion densities (3.20, 4.41), and 'SCRN' refers to screening factors (4.43).

For further details concerning the function or use of this machinery, the individual subroutine description in Chapter 100 should be consulted. In addition, the subroutine COPROD uses all of this machinery and may be consulted for an example.

5.22 VOLUME ELEMENT MACHINERY

To illustrate NTERAK's volume element machinery we will describe its use by the subroutine COPROD. COPROD's function is to generate the $\underset{\approx}{M} \cdot \underset{\sim}{U}$ product and $\underset{\sim}{U} \cdot \underset{\approx}{M} \cdot \underset{\sim}{U}$ inner product (4.30). Once slice information has been accessed and resides in core (5.21), COPROD must calculate residuals, element by element, and return the vector so calculated back to the disk in slices. These operations are complex and require fairly elaborate machinery. We begin by offering a brief overview.

COPROD begins by reading in the relevant slice information, establishing a section of the computational space in core. This volume is swept, element by element. Each element is characterized by the coordinates of its lowest indexed vertex. The potentials, and other vector information, for each of the eight vertices of a particular element are extracted from the main /CUBF/ array by VERTIO. VERTIO also replaces or augments array entries with calculated vertex information.

The potentials at the boundary of the computational space are assumed to be fixed and known (presently set to zero). Hence they are not stored explicitly in CBUF. Instead, COPROD examines each vertex of each volume element and determines if any be an implicit boundary. Those that do are fixed at the boundary potential. VERTIO (5.21, Appendix B) takes the X-Y displacement of the slices into account in picking out the vertices. Having set the potentials at the vertices of a particular element with VERTIO, COPROD calls ELEMNT to look up its characteristics in the list 'LTBL'. LTBL is an array with an entry for each element in the object grid, 'LTBL'. LTBL is an array with an entry for each element in the object grid, containing the following bit coded information (5.23): the element type, the number of surface cells sharing one or more nodes with the element (NCELLS), the element orientation, and the top/bottom flag.

The number of surface cells (NCELLS) sharing nodes with the element is used to refer to a second list, LCEL. If NCELLS is non-zero, DCVCEL is called to decode the next (NCELLS + 1) entries in the LCEL (5.25) list. The first word of this group will be an element I.D. number used merely as a check. The remaining words are also bit coded with: the surface cell number, the NODCOD telling which nodes are shared with the surface, and the FCN number (4.21) telling which element face, if any, the surface occupies.

If a vertex or node is shared by a surface cell, its potential is replaced with the surface potential for that cell, stored in the array SVRFV, sequentially by cell number. In the same way the contribution to the residual derived from the shared vertex is returned to the list SVRFR rather than the residual vector. Hence the surface potentialsd play the part of additional grid points, or variables in the matrix conjugate gradient equations. If a vertex is shared by more than one surface cell, the adjoining cell potentials are averaged by FORFIL (4.21) and assigned temporarily to the vertex. Once the vertice potentials have been collected, the residuals (4.30) are calculated by either PCUBES or ECUBES (4.20, Appendix B) and shared back to surface cells SRFR entry.

5.23 ELEMENT TABLE, LTBL

The element table, LTBL, is a list with one entry for every element in the grid. An element's LTBL entry can be accessed by calculating its relative address as if it were a 3-D array LTBL (X,Y,Z) where the Z range starts with the first slice in core (5.21).

Each word is coded in the following manner:

```
31 |0 9 8| |7 6 5 4 3 2 1 0 9| |8 7 6 5 4 3| |2 1 0|
     D              C                B          A
```

where:

A - The element-type number.

B - The number of surface cells sharing nodes with the element (up to 15) (NCELLS).

C - The orientation of the cell. This is only significant for partially filled cells and is explained in Reference 5-1.

D - Top/Bottom flag for elements above or below a thin plate.

D = 1 - bottom
= 2 - top
= 3 - both top and bottom, type 5

The following element types are allowed:

Type Number	Bits	Description
0	000	Empty cube
1	001	Half-empty wedge
2	010	Cube with diagonal on one face
3	011	Tetrahedron
4	100	Truncated cube
5	101	Empty cell bisected by a diagonal thin plate
6	110	Unused
7	111	Filled cube

These volume elements are described in 4.21.

5.24 SURFACE CELLS

5.24.1 SURFACE CELL LIST, KSURF

One of the lists produced by VEHICL and stored on file 11 (see Chapter 5.30 on file 11) is the surface cell list, KSURF. This list consists of bit packed word pairs for each surface cell. The information content is explained below, and the bit coding convention is illustrated in Figure 5.24/1. The bit ordering notation used here assigns each bit the exponent of 2 required to move an integer 1 to that location by multiplication.

COND = Conductor number, 1 to 15.

NORM = Surface normal in Miller indices, two bits for each index. The lowest bit representing 1 or 0, and the highest set for minus, off for plus.

XI,YI,ZI
XO,YO,ZO = Lowest coordinates of the volume element with which the surface cell is associated (the element that the cell normal points in to).
XI,YI,ZI

MAT = Material number, assigned by order of definition.

B = Set for all right-triangular surface cells (100,0-10, etc.) and for all equilateral (111) triangles whose enclosing volume element is mostly empty.

H = Orientation code for right-triangle surface cells. The two bits define the location of the right-angled corner in the plane of the triangle, i, j. The greatest bit refers to j. The j and i are related to the normal direction as follows:

	NORM	i	j
(0 0 ±1) =	Z	X	Y
	X	Y	Z
	Y	Z	X

XO,YO,ZO = Volume element that the surface normal points out of.

TB = 1 - bottom surface of a thin plate
= 2 - top
= 0 - N/A

The KSURF list is written on file 11 by either subroutine V.RESURF or O.RESURF. Prior to output it is 'Z' ordered to produce sliceability in the chosen orientation. It is written on 11 in a series of records with each record containing all the word pairs with a given ZI coordinate. These records are indexed by the NWSURF(2) list in record 1 of file 11. For each ZI of the object grid there is a NWSURF entry set equal to the number of KSURF words in the record for that ZI. If an NWSURF entry is zero, ther corresponding null 2C record of KSURF is skipped.

BIT	KSURF(1,N)		KSURF(2,N)
0 1 2 3 4 5	COND		MAT
6 7 8	NORM	ZN	B
9 10 11		YN XN	H
12 13 14 15 16 17	XI		XO
18 19 20 21 22 23	YI		YO
24 25 26 27 28 29	ZI		ZO
30 31 32			TB

Figure 5.24/1. KSURF surface cell list bit code.

5.25 LCEL, CONNECTIVITY

The LCEL list is used in the potential solving segment to connect node point to surface cells (4.20, 5.22). This is a bit packed list with the following structure:

ELT #
CODED WORD
CPDED WORD
.
.
.
ELT #

The ELT # is a coded element identifier,

ELT # = 4096 x (I + 64 * J + 4096 * K)

which serves as a check for correct addressing in the list. Addressing is accomplished by accumulating the NCELLS word count from the element table LTBL (5.22, 5.23).

The coded words are bit packed as follows:

36 ..31	0	9 8 7	6 5 4 3 2 1 0 9 8 7 6 5 4 3 2	1 0 9	8	7 6 5 4 3 2 1 0
		D	C	B		A

where:

A - Bits 0-7 are set if the corresponding vertex is shared with the surface cell. The vertices are numbered with X changing faster than Y which changes faster than Z, e.g., for element 1, 1, 1

	Coordinates		
Number	X	Y	Z
1	1	1	1
2	2	1	1
3	1	2	1
4	2	2	1
5	1	1	2
6	2	1	2
7	1	2	2
8	2	2	2

B - Is the standard orientation surface node number (see 4.21) that the surface will have when forming the element into which this surface points.

C - Is the surface number.

D - Is a code for surfaces that are on the top or bottom of a thin plate (4.21.1).

5.30 FILE SYSTEM

One of the most outstanding features of POLAR is its file management system. This system, combined with the sliced grid system (5.20) allows POLAR to model variably large 3-D problems (20,000 grid points is common), with fewer than 100,000 words of core. The modules VEHICL, ORIENT, NTERAK, and SHONTL communicate through the mass storage files 11 and 19 (see 5.31). Internal to each module, other files of differing types are used as scratch files. VEHICL uses the NASCAP object definition coding and thus a variety of NASCAP files are used initially in VEHICL as scratch files. This document will ultimately describe these files, but the reader is presently referred to Reference 1. Two files (11 and 19) are used instead of just one, so that one may be indexed with literal name keys (19), and the other (11), with number type index keys.

5.31 MASS STORAGE FILE MANAGEMENT

Data stored on the permanent file 11 and 19, and the scratch file 9 and 10, is written and retrieved using the CDC MSIO system (mass-storage-input-output). On non-CDC machines, the MSIO routines OPENMS, WRITMS, READMS, and CLOSMS are provided by POLAR FORTRAN routines which mimic the CDC versions to provide a machine independent interface to the host system. We now describe these four routines as they are used in POLAR.

OPENMS:

Example: DIMENSION IND (NUM)
CALL OPENMS (LUN, IND, NUM, t)

LUN - The logical unit number of the file.

IND - Space provided for a working copy of the file index.

NUM - Length (words) of IND

t - type of index
= 0, number type index
= 1, name type index
To index XR records, a t = 0 file requires NUM $\geq$ XR + 1; for t = 1, NUM $\geq$ 2*XR + 1.

READMS:

Example: DIMENSION BUFR(1000)
CALL READMS (LUN, BUFR(100),NWDS,KEY)

LUN - Logical unit number

BUFR(100) - Starting address to which data is to be read

NWDS - Number of words to be read

KEY - The index key under which the data record was stored.

5.30 FILE SYSTEM

One of the most outstanding features of POLAR is its file management system. This system, combined with the sliced grid system (5.20) allows POLAR to model variably large 3-D problems (20,000 grid points is common), with fewer than 100,000 words of core. The modules VEHICL, ORIENT, NTERAK, and SHONTL communicate through the mass storage files 11 and 19 (see 5.31). Internal to each module, other files of differing types are used as scratch files. VEHICL uses the NASCAP object definition coding and thus a variety of NASCAP files are used initially in VEHICL as scratch files. This document will ultimately describe these files, but the reader is presently referred to Reference 1. Two files (11 and 19) are used instead of just one, so that one may be indexed with literal name keys (19), and the other (11), with number type index keys.

5.31 MASS STORAGE FILE MANAGEMENT

Data stored on the permanent file 11 and 19, and the scratch file 9 and 10, is written and retrieved using the CDC MSIO system (mass-storage-input-output). On non-CDC machines, the MSIO routines OPENMS, WRITMS, READMS, and CLOSMS are provided by POLAR FORTRAN routines which mimic the CDC versions to provide a machine independent interface to the host system. We now describe these four routines as they are used in POLAR.

OPENMS:

Example: DIMENSION IND (NUM)
CALL OPENMS (LUN, IND, NUM, t)

LUN - The logical unit number of the file.

IND - Space provided for a working copy of the file index.

NUM - Length (words) of IND

t - type of index
= 0, number type index
= 1, name type index
To index XR records, a t = 0 file requires NUM $\geq$ XR + 1;
for t = 1, NUM $\geq$ 2*XR + 1.

READMS:

Example: DIMENSION BUFR(1000)
CALL READMS (LUN, BUFR(100),NWDS,KEY)

LUN - Logical unit number

BUFR(100) - Starting address to which data is to be read

NWDS - Number of words to be read

KEY - The index key under which the data record was stored.

WRITMS:

Example: CALL WRITMS (LUN,BUFR,NWDS,KEY,r)

LUN, NWDS, and KEY are as for READMS; data is read from BUFR. r = -1, always for POLAR. This specifies that a record may be overwritten only if the new record length does not exceed the old length. When this occurs, a new record is written and a link stored in the old location.

CLOSMS:

Example: CALL CLOSMS (LUN)

This routine writes the working index from core to the file copy. It is called frequently to maintain an up to date file copy of the index to protect data against unexpected crashes and stops.

5.32 MRBUF PLUS BUFSET PLUS FRIENDS

To enhance the efficiency of both the execution and development of POLAR, a formalized system of variable characterization and identification is used. This begins with the array VPROPS(20,70) wherein 20 different properties of 70 different variables are initialized by MRBUF. To set up a work space for a variable (or a slice of a variable) in the general purpose buffer CBUF, a call is made to BUFSET using the literal name of a variable (5.33). BUFSET will move the vector properties from VPROPS to a working array (1AVECS(20,I)) and the index I will be written into the MELAD common block in the variable pseudonym. For example

CALL BUFSET(5, 'POT', ...)

would result in space for five Z slices of potential to be allocated in CBUF with all essential information stored in IAVECS (20,IPOT) with IPOT being found in the MELAD common block.

The IAVECS properties are listed below, with a V indicating a non-variable property that is placed in VPROPS by MRBUF.

V	IAVECS(1,I)	=	Name
	IAVECS(2,I)	=	Start address in CBUF
	IAVECS(3,I)	=	Number of slices
	IAVECS(4,I)	=	Vector length of a slice
	IAVECS(5,I)	=	Words per vector

(The number of words in a slice is IAVECS(4,I) * IAVECS(5,I).)

V	IAVECS(6,I)	=	Mass storage file number
	IAVECS(7,I)	=	Low pointer (shifted slice coordinates), see property 14 and Section 5.21
	and		
	IAVECS(8,I)	=	High pointer used by the data paging routine PAGER
	IAVECS(9,I)	=	Lower (object coordinates, 5.21)
	and		
	IAVECS(10,I)	=	Upper object limits, inclusive
	IAVECS(11,I)	=	Starting key number for the number key files.
V	IAVECS(12,I)	=	X dimension of a slice
V	IAVECS(13,I)	=	Type of slice or vector. Choices are: 'VARI' - variable length (example, LCEL) 'SNGL' - single slice (example, SRFV) 'WORK' - work space 'OTHR' - other (FOTO) 1 - object slice (LTBLO 2 - real grid (POT) 3 - outer virtual node (LCOF) 4 - virtual element grid (DION)
V	IAVECS(14,I)	=	Shift added to slice coordinates to prevent FORTRAN MOD function of negative or zero slice coordinates
V	IAVECS(15,I)	=	Y dimension of slice

V IAVECS(16,I) = Lower Z limit of the variable range

IAVECS(17,I) = Pointer to relate node data to auxilliary node data (necessary for double points created by thin plates)

V IAVECS(18,I) = Word type, 'REAL', 'INIT' (integer), 'OCTL', 'ALPH'

5.33 MRBUF VARIABLE LIST

NAME	DESCRIPTION (See Section)
LTBL	Element table
KSURF	Surface cell list
SRFR	Surface residual list
SRFV	Surface voltage list
LCEL	Surface to node connectivity list
POT	Node potentials
GI	Neutral ion densities
SCRN	Linearlized screening term
MU	Product of matrix $\underset{\approx}{M}$ dot $\underset{\sim}{U}$
R	Node residuals
DD	*Unused*
U	Congugate gradient error vector
RHOI	Sheath ion densities
QE	Average element electric field
LTYP	Element location with respect to sheath. *Also a calculation saver used by CURREN.*
LCOF	Table to provide direct addressing to the LCEL list by element address
SRFV	Surface V list
SRFY	Surface Y list
VMAP	Velocity space map, work space
RHS	Right hand side of charging equation
JTOT	Total surface current list for insulators and conductor list
JSM	Small surface currents for insulators and conductor list
VMC	Voltage vector used to construct RHS (voltage minus conductor)
IFIX	List indicating surfaces with fixed voltages in insulator plus conduct list form
RMAT	The capacitance matrix portion of charging equation

NLST	A list of insulating surfaces
LIST	Defines element location ICCG packed sparse matrix
MRKR	Marks the starts of new rows in LIST
AMAT	A work space used for calculating RMAT
VDT	Change in voltage portion of charging equations (the answer found by ICCG)
SCRT	Scratch vectors
CLRG	Vector list of large capacitances of insulating surfaces
CSML	Vector list of small capacitances to insulating surfaces and conductors
JPRM	List per surface of derivative of total current by surface voltage
SGMA	Conductivity matrix including grid size
TSRV	Trial surface voltage list
SREL	List of pointers per surface into the LCEL list
SRFC	Weighted sums of particles which impacted on object during CURREN
SRFI	List of surface ion currents
TKSR	Sets CBUF space to hold all KSURFs at once
SRIA	Estimate of ambient ion surface current
ASRF	All of KSURF in CBUF, unsliced
JFIX	Similar to IFIX only for each surface cell
VXB	V cross B voltage bias for each insulating surface
SRDT	Keeps the voltage of fixed surfaces
EXTV	Surface voltages found during first trial step
JTSR	Total current to each surface
JSMS	Total of "small" currents to each surface
SPAC	Work space
IMAT	Material type of each surface

ICND	Conductor number of each surface
SFAR	Area of each surface
OLDV	Voltage from previous charge step
DION	Composite ion density array
FOTO	Memory of past charge information (not implemented yet)
SUMM	Code history information, summary information from previous PWASON, CURREN and CHARGE cycles (not implemented yet)
QUSD	Stabilized charge density array
AXGI	Auxiliary GI values
AXDI	Auxiliary DION values
AXSC	Auxiliary SCRN values
AXQU	Auxiliary QUSD values
AXRI	Auxiliary RHOI values
LAUX	Coordinate list for aulilary values
PSGM	Pre sigma (conductance) calculated by VEHICL and DRIENT. Does not include grid size.
OLJT	Previous CHARGE cycle total surface currents from first trial step
OLSV	Surface voltages used to find OLJT

5.60 ION DENSITIES

The calculation of ion charge densities is performed in two major segments. Before any electric field information is available the straight line thermal ion density is calculated everywhere in the grid. When potentials have been calculated, the sheath edge is found and particles are tracked from the sheath edge to the object. The algorithms for these two calculations are described in the following sections.

5.61 PRESHEATH IONS

The presheath ion density calculation consists of determining the function $g(\underset{\sim}{x},\Omega)$ and described in Section 3.2. This function has value unity when particles can reach point $\underset{\sim}{x}$ from angle Ω without hitting the vehicle, and it has value zero when the particles are prevented from reaching $\underset{\sim}{x}$ because they are blocked by the object. The major routine in this section of the code is GEMFAC for geometrical factor, and the results used by NTERAK are GI's for geometrical ions. The GI's are element centered and defined as

$$GI(I,J,K) = \frac{1}{n_{0_i}} \int g(\underset{\sim}{x},\Omega) \int f_{i0}(r,v) v^2 \, dv \, d\Omega$$

For calculation of these geometrical factors we work in a polar coordinate system whose $\theta = 0$, $\phi = 0$ ray points into the oncoming ions. In this system, the function f_{i0} is azimuthally symmetric and the integral is reduced to

$$GI(I,J,K) = \sum_{\theta_i} f(\theta_i) \left(\sum_{\phi_i} g(\theta_i,\phi_i) \Delta\phi_i \right) \Delta\theta_i$$

where the function $f(\theta)$ is normalized so that

$$\sum_{\theta_i} \sum_{\phi_i} f(\theta_i)\, \Delta\theta_i\, \Delta\phi_i = 1$$

This function is calculated in FCAL. The calculations are done on a uniform grid with a default resolution of 1° in θ and 10° in ϕ. The greater resolution in the polar angle θ is necessary since f changes so rapidly as we move away from the ram direction. The geometrical factor is calculated by taking the building block surfaces (A2's from HIDCEL) and projecting them onto the θ, ϕ plane. Regions inside of each surface are then marked as excluded. The major source of error is the interpolation between vertices in θ, ϕ space, since the straight line edges in Cartesian space are not straight lines in θ, ϕ space. To improve accuracy, ADVERT adds extra vertices in the middle of edges for each A2 surface. The number of vertices per edge is controlled by NADD, and has default value of 2. The calculations to determine the points inside the surfaces are usually done in STICK (for STICK a one into the GEM array). However, the cases of the ram or anti-ram directions being excluded must be treated separately. The poles in the θ, ϕ space are singular points and are handled in STKUP and STKDN for the $\theta = 0$ and $\theta = \pi$ poles, respectively.

GEMFAC is called for each surface by NEUDEN (for neutral approximation density) which also performs the angle summations. The principle advantage of this geometrical technique is speed, the tens of millions of $g(\theta,\phi)$ needed for the densities at each grid point can be calculated in just a few minutes of computer time. A few representative plots of ram, regular and anti-ram polar angle projections are shown in Figures 5.61/1-5.61/3.

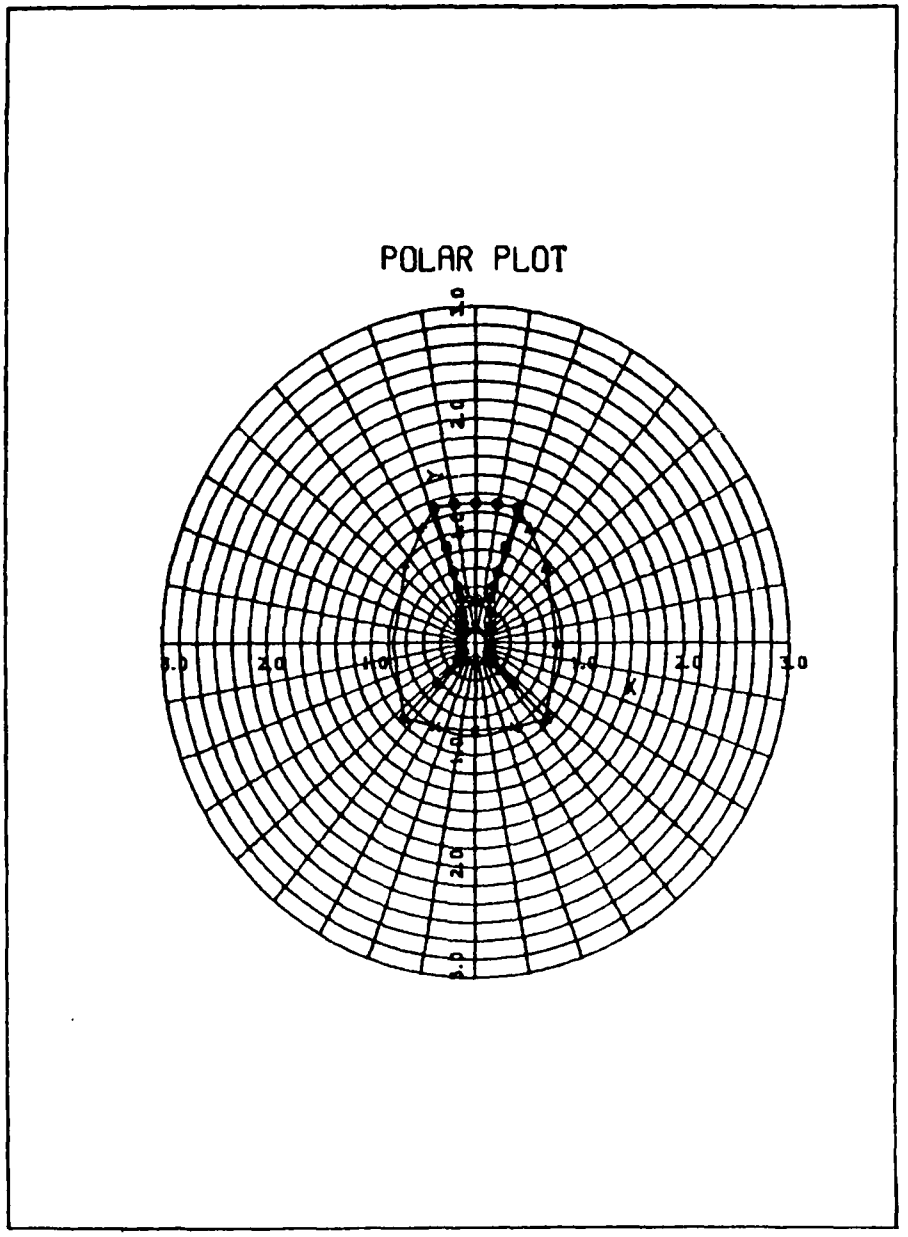


Figure 5.61/1. Neutral approximation phase space map of a flying brick blocking the ram direction.

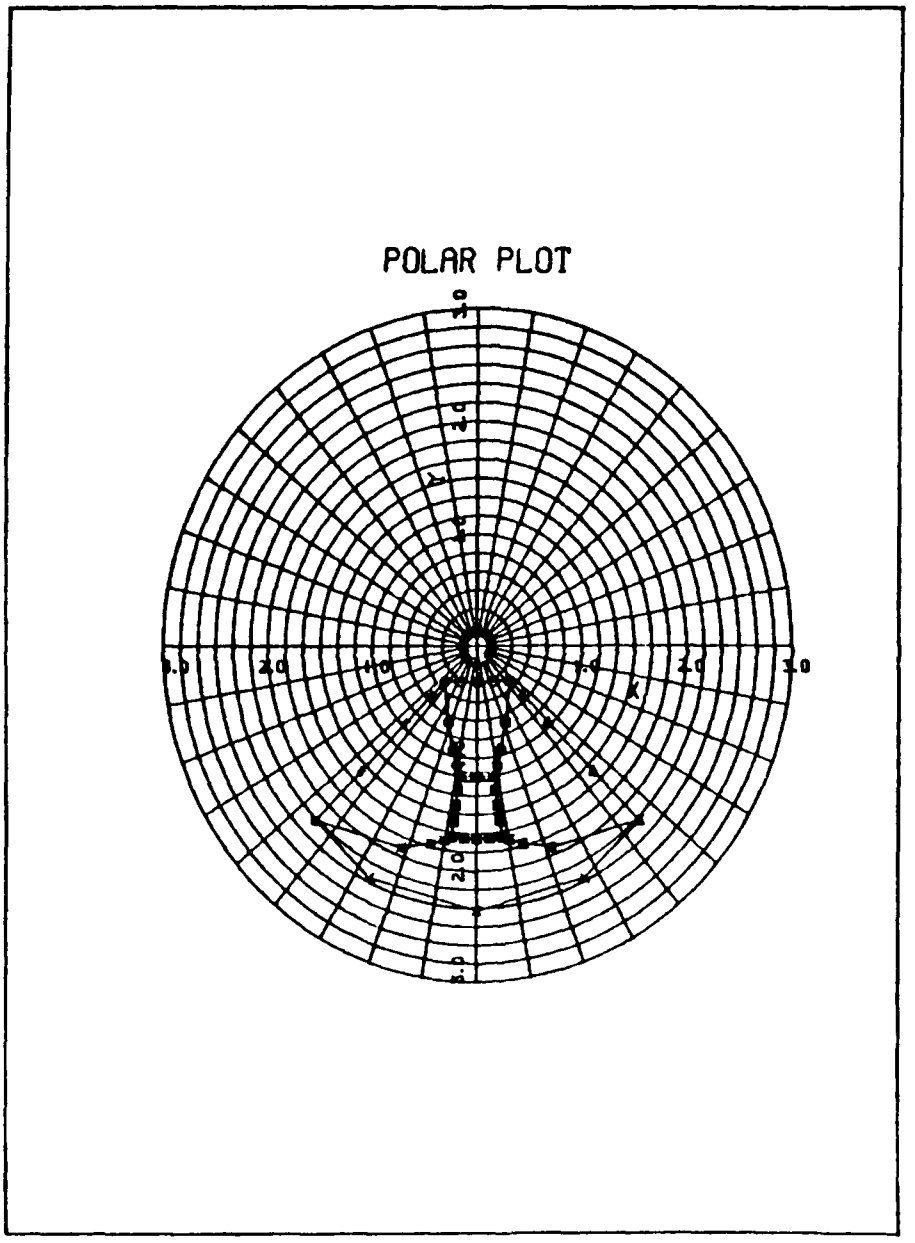


Figure 5.61/2. Neutral approximation phase space map of a flying brick at right angles to the ram direction.

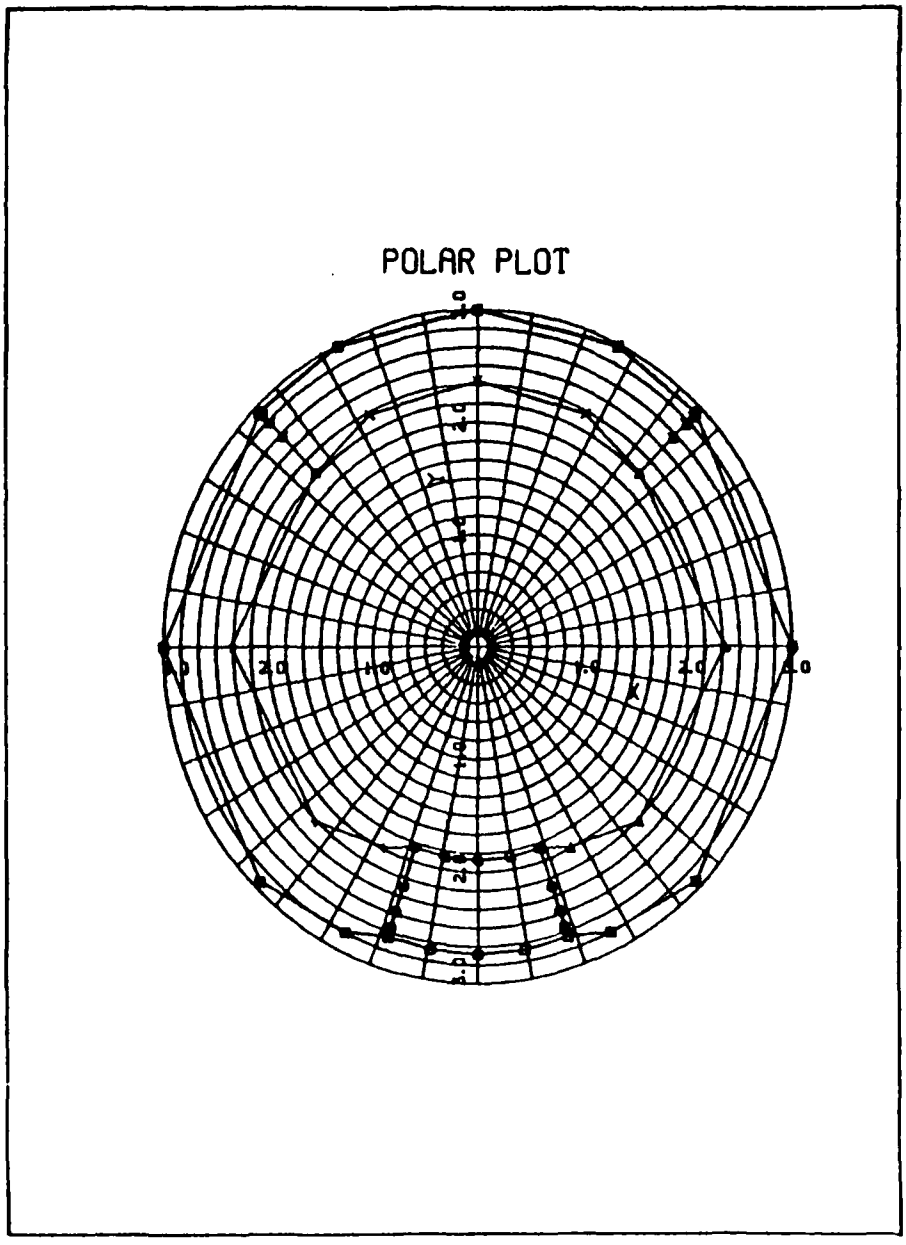


Figure 5.61/3. Flying brick in the anti-ram direction.

5.62 SHEATH IONS

The space charge density of ions internal to the sheath boundary, and currents of ions to the vehicle surface must be determined by particle tracking. The concepts involved were discussed in Section 3.30 and 4.42. This section documents the coding responsible for these calculations, but a brief overview is in order.

External to the sheath, electric fields are weak enough to allow for accurate estimates of the ion flux to any portion of the sheath surface, including ram effects. Presuming a previous Poisson calculation, the sheath is currently defined to be an equipotential near the ambient plasma temperature (this is an input parameter). Once the sheath is located, it is divided into subareas. These subareas subsequently become 'particles' which represent flux tubes of constant current, rather than constant charge. This current, referred to as current weights, is the subarea times the input flux density of ions (Section 4.42) to the subarea. The actual particle tracking is done per slice (4.11) with the particle 'pusher' sweeping the grid alternately in the +Z (called right) and -Z (left) direction. Trajectories evolve in the X and Y coordinate directions until they exit a slice whereupon the trajectory information is stored. The trajectory is picked up in the next slice or the return pass of the pusher. Ultimately, they leave the problem or more commonly, hit a surface and are moved to a 'dead-particle' list, and are later processed into surface currents.

Throughout all of this pushing, the time a particle spends in a volume element is multiplied by its current to determine the space charge contribution to the element.

5.62.10 SHEATH EDGE

The routine STHCAL finds the sheath boundary, divides it into "particles", and then writes sliced lists of particles onto file 10. The boundary potential was calculated by a routine called FNDBND. Currently, FNDBND finds the boundary potential by asking the user to supply it. Later, this routine will calculate a potential.

Once the boundary potential is known, STHCAL loops through all of the elements, calling SHEATH to check each element to see if it contains the sheath potential. If part of a boundary passes through a cell, it divides the boundary surface into triangular particles. The particles are defined by their area (in grid lengths squared) and the location of their centroid.

Sometimes SHEATH finds particles which cannot exist. For example, in large objects some of the nodes within the object will have potentials of zero. If the surface potential is higher than the sheath potential, SHEATH will think it found part of the sheath boundary and define some particles. It is also possible to find particles which would have had to come from a non-source region. These particles are eliminated by backtracking to see if there is a surface or a high potential region which would have blocked the particle.

Once all the particles in a z-slice (4.11) are defined, they are written into file 10 with their area, the location of their centroid, and their initial velocity. Presently, the initial velocity is defined to be zero. Then the particle lists are stored in a linked list data structure (5.62.11) in chunks of 100 or fewer particles.

5.62.11 THE PARTICLE LIST STRUCTURE

The particle list is kept in a linked list data structure. An array in the common block /LNKCOM/ contains either -1 or the key of a particle list. The key is key number of the particle list in file 10 as it is used by the MSIO package.

To find the key of the first particle list of a certain z-slice, the /LNKCOM/ array LKPART(i) (LKSCUR(i) for surface currents) is indexed by the z-coordinate of the slice (4.11, 4.12). The rest of particle lists for a given z-coordinate are indexed by the first word from the preceding list. This is the MSIO key of the next slice. When a particle list does not have another list following it, the key it holds is a -1. The negative one is a flag signalling the end of a linked list.

For example, if our problem had a z-coordinate which varied from 0 to 6 and 200 particles in z-slice 0, 730 particles in slice 3 and 1300 particles in slice 4, file 10 would be similar to table 5.62.11/1. Since the order in which the particles were stored affects the actual contents of the table, there are a number of possible arrangements.

In the table, 230 particles of the z = 3 slice are in file 10 at key 1 and 500 are at key 5. Since next key is -1 for key 5, we know there are no more particles in this slice.

As particles are moved to other slices, space opens up in the middle of the file (keys 3, 4 and 8 in the example). To conserve space and to keep the file packed, the emptied keys are also linked. So when a new particle list needs to be stored on file 10, an emptied key will be used if there are any available. The key of the first empty location is stored in the variable 10LOKY.

TABLE 5.62.11/1
AN EXAMPLE OF THE PARTICLE LIST DATA STRUCTURE

First Key List:

		LKPART(i)						
	ioldky	i = 0	1	2	3	4	5	6
key =	4	6	-1	-1	1	2	-1	-1

file 10 contents

Key No.	Next Key	No. Particles	z-slice
1	5	230	3
2	9	300	4
3	-1	0	-
4	8	0	-
5	-1	500	3
6	-1	200	0
7	-1	500	-
8	3	0	-
9	7	500	4
10	empty	empty	-

Note: This example assumes a maximum page size of 500 particles.

To store the surface currents, another array LKSCUR(i) is used to hold the key of the first particle list for each z-slice. The linked lists allow the surface currents to use the empty spaces left by the particle list used by the sheath particle tracking section, which helps keep the file packed.

5.62.12 PARTICLE PUSHING UNITS

It is convenient to define special units for the trajectory calculations. The time is in units of the ion acoustic period, the distance is in grid lengths, the velocities are in Mach velocities, and the acceleration is unitless. The quantities are defined by

$$t(\text{grid}) = t(\text{sec}) * \frac{v_s \text{ (ion acoustic speed)}}{h \text{ (grid length in meters)}}$$

$$x(\text{grid}) = x(\text{meters})/h(\text{grid length in meters})$$

$$v(\text{grid}) = v(\text{meters/sec})/v_s(\text{ion acoustic speed})$$

$$a(\text{grid}) = QE = \frac{E_v(\text{electric field in volts}}{T_v(\text{ion temperature in volts})}$$

where the ion acoustic speed is

$$v_s = \sqrt{\frac{kT}{m}} \text{ k-Boltzman, T °K, m - kg.}$$

After MOVPAR moves the particle, it increments RHOI (space current) by the current of the particle times the time it was in the cell.

5.62.20 SHEATH CURRENT

The routine CURREN controls the process of finding the sheath current (see Figure 5.62.20/1). CURREN calls the initialization and exit routines CURPEP and CUEXIT, respectively, and it pushes the particles alternately to the right (+Z) then left (-Z) until all of the particles have left the grid or hit the object. The variable, NPRTCL, is the number of particles left which have not been pushed to completion.

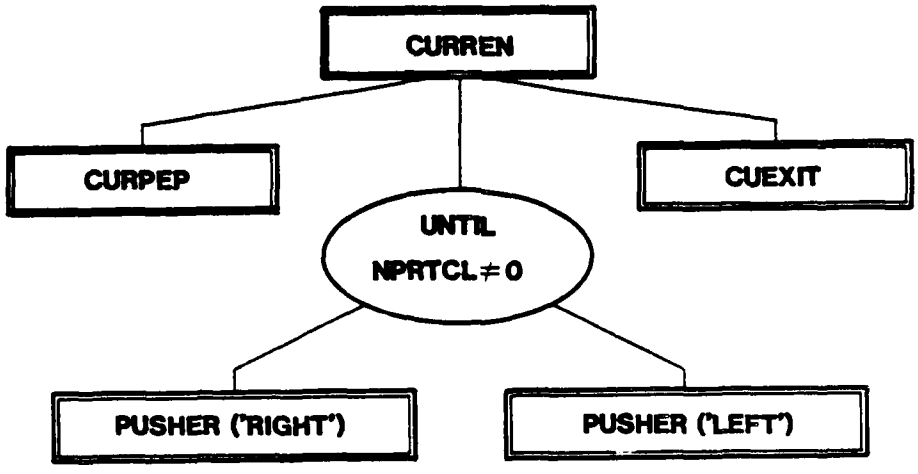


Figure 5.62.20/1. Structure diagram of the subroutine CURREN.

5.62.21 CURPEP (CURRENT PREPARER)

Called by CURREN, CURPEP initializes the particle counters, opens file 10 and initializes it (5.62.11), clears CBUF with a call to the routine BUFCLR, and calls STHCAL to find the initial set of particles.

Once the initial particle lists are calculated, CURPEP resets CBUF for PUSHER with calls to BUFCLR and BUFSET. Then the force table (QE), the extended element table (LTYP), and the sheath current table (RHOI) are initialized to complete the routine. To speed up the calculations in XITCEL (5.62.22), the force table (QE) contains the acceleration. The acceleration is equal to the negative trilinear electric field divided by the temperature in volts. The units of the acceleration are grid units (see 5.62.12).

5.62.22 PUSHER (PARTICLE PUSHING)

This routine pushes all the particles to the left or to the right. PUSHER loops through all of the particles in each z-slice, pushing particles until they leave that slice. The particles leaving in the forward direction are pushed again in the next slice and so on when a particle's path intersects an object face, the particle is transferred to the surface current list (5.62-11) and counted as a dead particle. Particles detected leaving the computational grid are counted and removed from the problem (although their previous trajectory information remains).

The routine which PUSHER calls to push a particle through a cell is named PUSH.

PUSH decides which of two pushing techniques to use to push a particle out of its cell. PUSH checks the element table (5.23) to see if the element is next to the object or in a partially filled cell (both are labeled 'NEAR' in the LTYP table in CBUF). The routine STPPSH moves particles which are 'NEAR' the object. If the particle is not close to the object, PUSH calls XITCEL (exit cell) to find the time the particle needs to exit the cell, and MOVER to move the particle to its new location.

XITCEL solves the following six quadratic equations for the time, t_{ji}, required for the trajectory to be traced from an entry coordinate X_{oi} to an exit coordinate X_{ji}:

$$X_{ji} = X_{oi} + V_{oi} t_{ji} + \frac{1}{2} QE_i \, t_{ji}^2 \quad \begin{cases} i = X, Y, Z \\ j = +, - \end{cases}$$

where V_{oi} is initial velocity. QE_i is the acceleration calculated from the electric field at the center of the element. The smallest positive real t_{ji} is chosen as the time required by the particle to leave the cell. See Section 5.62.12 for a discussion of the special units used for the particle pushing.

MOVER used the time found by XITCEL to find the new particle location. First it calculates the new position and velocity using

$$X_i = X\phi_i + V\phi_i t + \frac{1}{2} QE_i t^2$$

$$V_i = V\phi_i + QE_i t$$

Then it shifts the particle position by one thousandth of the velocity, moving forward along the exit axis, found by XITCEL, and backwards along the other two. This is necessary because the particle's cell number is referenced by taking the integer portion of the particle position. If a particle stopped exactly on the top side of element (i,j,k) as it moved downward, the address found by truncation would be (i,j,k+1). To prevent this form of faulty addressing, the particle is moved off of the cell boundaries.

Because the central electric field is used for the entire element, and the total energy may drift with each move, MOVER renormalizes the velocity vector to force the particle to conserve energy. After MOVER pushes the particle out of the element, PUSH increments the RHOI (space current density) by the time spent in the cell multiplied by the area of the particle.

When the particle gets closer to the object, MOVER can no longer be used since the particle could strike an object surface before it left the cell. To push particles within a grid length of the spacecraft, PUSH calls STPPSH (step push). STPPSH checks to see if the particle is inside a filled portion of an element. If it is, the routine finds which surface it passed through and returns to PUSH. If not, STPPSH calls STPPAR (step particle) which moves a particle for time t. Time t is the smaller of the time required for the particle to free fall or move at a constant velocity a distance of one-tenth of a grid length.

$$t = \text{smaller of } \left(D/|\tilde{V}|, \sqrt{\frac{2D}{\tilde{E}/\theta)}} \right)$$

where $\vec{E}$ is the electric field at the particles position, θ is the plasma temperature and D is one-tenth of a grid length. When the time step has been computed, the particle is moved to X_i, (i = x,y,z)

$$X_i = X_{oi} + V_{oi}t + \frac{1}{2}\frac{E_i}{\theta}t^2$$

$$v_i = v\phi_i + (E_i/\theta)t$$

After the particle is moved, STPPSH increments the RHOI (space current density) by the current weight of the particle times the time it was moved. Then checks to see if the particle left the cell by counting how many of the element coordinates of the new position differ from the old. If the particle did not leave the cell, energy conservation is checked and the velocity is renormalized if there is more than a $\pm$5 percent error. Then the particle is pushed again.

If the particle did leave the cell, energy conservation and the number of element coordinates which have changed are checked. If the number is greater than one or the energy is off, the particle is backed up until only one element boundary is crossed and the energy is right. The routine that does this is called MOVBAK (move back). It is important to keep particles from crossing more than one boundary at a time, since it is possible for particles to miss corners of objects.

MOVBAK backs up a particle by finding the average velocity of a particle during the timestep ($\vec{v}_{ave} = (\vec{v}_\phi + \vec{v})/2$) then solving $\vec{x} = \vec{x}_\phi + \vec{v}_{ave}t$ for t on all of the sides of a cube. Using the smallest positive time it finds, it moves the particle from its original point to $\vec{x}'$.

$$\vec{x}' = \vec{x}_\phi + \vec{v}_{ave}\, t_{min}$$

MOVBAK then moves the particle off of the cell boundary by shifting the position by 0.001 times the velocity in the same manner as MOVER did previously.

After calling MOVBAK, STPPSH checks energy conservation again. After renormalizing the velocity, if necessary, and incrementing RHOI, the particle is checked to see if it hit a surface as it left the element.

5.62.23 CUEXIT (CURRENT EXIT ROUTINE)

This routine takes care of the exit from CURREN. It prints a tally of what happened to the particles from SHEDGE and it closes file 10, which now contains the surface currents.

5.71 ION SURFACE CURRENTS

The ion surface currents are found in the list, SRFI. This list is created in the segment headed by the subroutine IONCUR from the dead particle list (dead-list) created by the CURREN segment. The upper structure of this segment is illustrated in Figure 5.71/1.

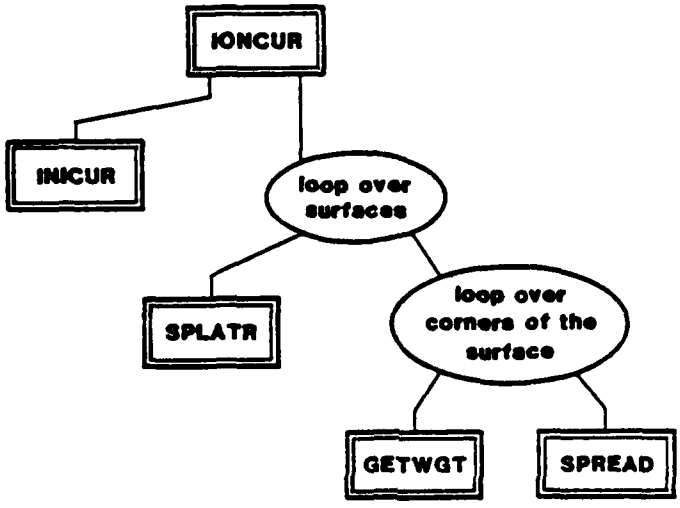


Figure 5.71.1.

The creation of the SRFI list occurs in two main steps; step 1 reduces the dead-list to an intermediate SRFC list, and step 2 redistributes the surface currents among the surfaces. The justification and computational techniques for these steps have been discussed in Section 4.53.

Step 1 is controlled from IONCUR and performed by INICUR. The dead-list can be very large with an order that is best described as chronological. A particular surface will appear as often as it is struck by a particle. To speed up the surface current calculation, and to reduce the noise inherent to particle pushing, INICUR reads through the dead-list sequentially and simultaneously accumulates in the SRFC list the "average" particle (see 4.53) for each surface.

This average particle has the sum of the contributing current weights with an average impact location, energy, and velocity. These averages are weighted by the current weights (4.53).

Step 2 is the redistribution or sharing of the "raw" surface currents in the SRFC list. This is done to further reduce the "noise" in the raw ion surface currents (4.53). Thus, IONCUR will next loop over the surfaces in SRFC and for each surface, S, call SPLATR to calculate the bilinear weights, SPW (node), which are used to distribute the SRFC current to the vertices of the surface. These node currents are to be shared back to all of the surfaces adjoining each node. IONCUR loops over the surface vertices calling GETWGT to calculate the CRW (node, ns) where ns indexes neighboring surfaces (4.53). The surface-node-surface connectivity is provided by the LCEL list (5.25). Since the LCEL list is designed to be read sequentially and is ordered by slice and element, direct access to the needed section of LCEL is provided by an index list SREL (5.25).

After the CRW have been obtained for a node, SPREAD is called to perform the summation;

$$SRFI(ns) = SRFI(ns) + UNITS * CRW(node, ns) * SPW(NODE) * SRFC(S)$$

where ns ranges over the neighboring surfaces including surface S. UNITS = ECHRG * DXMESH^2 * FNORMI, where ECHRG = 1.6×10^{-19} Coulombs and DXMESH is the length of a mesh unit in meters. FNORMI is the unperturbed Maxwellian flux density against which the current weights are calibrated;

$$FNORMI = n_i \sqrt{kT/2\pi m_i}$$

where, in MKS, n_i is the ion density, k is Boltzmann's constant, T is temperature and m_i is the ion mass.

5.73 CHARGE (SURFACE CHARGING CONTROL)

CHARGE is the entry level routine into the surface charging section of POLAR (see Figure 5.73/1). It initializes the charge cycle by reading in the material properties and calculating the plasma to surface (routine MAKCSM) and surface to conductor (MAKCLG) capacitances.

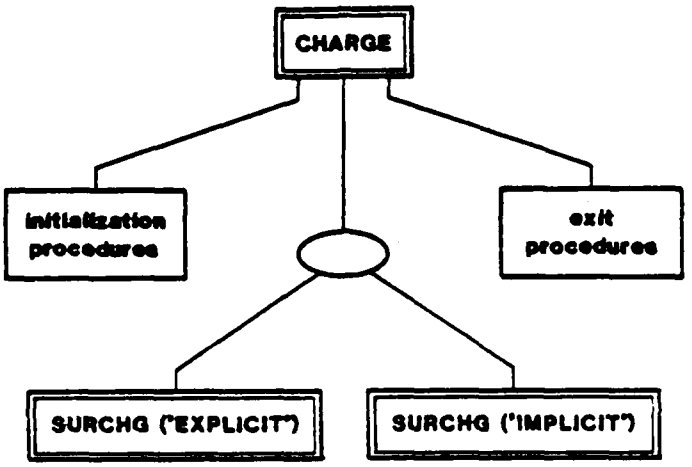


Figure 5.73/1. Structure diagram of subroutine CHARGE.

After the initialization routines have been called, CHARGE calculates the explicit then implicit surface potentials. Presently, CHARGE loops on the SURCHG sequence by the fixed input number, MAXITT; however, a convergence test will be placed in the location indicated by the empty oval. Upon conclusion, the array SURFV contains the new surface potential which is stored on file 19.

The computational theory for the charging section is discussed in Section 4.50.

5.73.1 SURCHG (SURFACE CHARGER)

SURCHG calls the routines needed to set up the equation to be solved and finds the next set of surface voltages. SURCHG can solve the charging equation in either the explicit form (Eq. (5.73-1-1)) or the implicit form (Eq. (5.73.1-2)).

$$\left(\frac{\underset{\sim}{C}}{\Delta t} + \underset{\sim}{\sigma}\right)\Delta \underset{\sim}{V}_t = \underset{\sim}{I}_t - \underset{\sim}{\underset{\sim}{\sigma}} \underset{\sim}{V}_t \qquad (5.73.1\text{-}1)$$

$$\left(\frac{\underset{\sim}{C}}{\Delta t} + \underset{\sim}{\sigma} - \frac{d\underset{\sim}{I}}{d\underset{\sim}{V}}\right)\Delta \underset{\sim}{V}_t = \underset{\sim}{I}_t - \underset{\sim}{\underset{\sim}{\sigma}} \underset{\sim}{V}_t \qquad (5.73.1\text{-}2)$$

Using a flag sent by CHARGE, SURCHG chooses which form of the charging equation to solve (see Figure 5.73.1/1). When solving the implicit charging equation (5.73.1-2), the SURCHG assumes an explicit solution had been found previously.

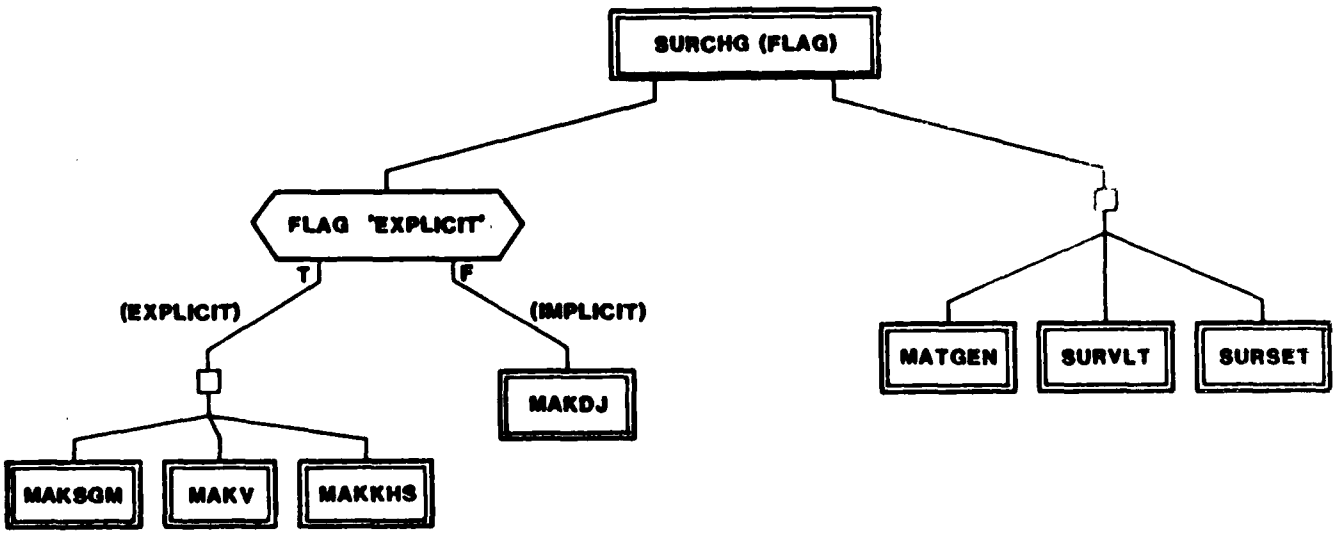


Figure 5.73.1/1. Structure diagram of the routine, SURCHG.

SURCHG, called with an "explicit" flag, generates $\underset{\approx}{\sigma}$(MAKSGM), $\underset{\sim}{V}(t)$ (MAKV), and $\underset{\sim}{I}(t) - \underset{\approx}{\sigma}\,\underset{\sim}{V}(t)$(MAKRHS), then calculates $\underset{\sim}{V}(t+\Delta t)$ explicitly. When an implicit flag is received, MAKDJ is called to calculate dI/dV. MAKDJ will call MAKJ which will recall the current routine (4.50, 5.70) for an estimate of $I(t+\Delta t)$ using the explicit $V(t+\Delta t)$.

At this point, both charging equations can be treated the same way. The matrix,

$$\frac{\underset{\approx}{C}}{\Delta t} + \underset{\approx}{\sigma} - \frac{d\underset{\sim}{I}}{d\underset{\sim}{V}} \quad ,$$

is calculated by MATGEN (using dI/dV = 0 for the explicit case), setting up $\underset{\approx}{M}$ for the ICCG call by SURVLT. SURVLT solves Eq. (5.73.1-3)

$$\underset{\approx}{M}\,\underset{\sim}{\Delta V} = \underset{\sim}{R} \qquad (5.73.1\text{-}3)$$

where $\underset{\sim}{R}$ is the vector found by MAKRHS and $\underset{\sim}{\Delta V}$ is the change in the surface potential.

Using the $\underset{\sim}{\Delta V}$ found by SURVLT, SURSET calculates the new surface potentials and saves them, completing the charging timestep. The following sections discuss the above routines in more detail.

5.73.2 CHARGING MATRIX AND VECTOR FORMULATION

As discussed in Section 4.51, there are two cases which affect the charging equations, 5.73.1-1 and 5.73.1-2, depending on the occurrence or non-occurrence of fixed potential surfaces. When a surface potential is fixed, the matrices and vectors of 5.73.1-2 are constructed exactly as implied. If a surface has been fixed by CBRAIN (called from SURCHG) SURCHG passes a flag, IFIXED, to the matrix and vector routines (MATGEN, MAKRHS, MAKJ, MAKDP, MAKV) to construct the more diagonalized matrices, $\underset{\approx}{C}$ and $\underset{\approx}{\sigma}$, and the corresponding vectors $\underset{\sim}{V}$ and $\underset{\sim}{I}$. These transformations are discussed in 4.51, but they are characterized by constructing $\underset{\sim}{V}(t)$ from the potential difference between a surface and the underlying conductor instead of simply the surface potential. The effect of this is to reduce the off-diagonal elements and increase the diagonal elements of the amalgamated matrix M (Eq. (5.73.1-3)). ICCG likes this and will generally converge faster.

Since there can be as many as 1200 surfaces or so, it was necessary to pack all the matrices in the problem (unpacked it could be 1 million words/matrix). Since the matrices are mostly empty, just saving the nonzero entries and their locations simplifies the problem greatly. The entry locations are stored in a list (called 'LIST' in MRBUF) with a second list ('MRKR') being used to indicate the location of rows in LIST. The entry locations are stored row by row with the first entry for a row being the negative row number. For example, the nonzero locations in the following matrix

a	0	0	0	0
0	a	b	0	0
0	b	a	e	f
0	c	e	a	d
0	0	f	d	a

would appear in LIST as -1, -2, 3, -3, 2, 4, 5, -4, 2, 3, 5, -5, 3, 4 and MRKR would be 1, 2, 4, 8, 12, 10000 with the 10000 signalling the end. Since the matrices always have nonzero diagonals, the -2 in LIST means there is an entry at (2, 2). The 3 following it means (2,3) and so on. For large problems this greatly reduces the data requirements.

MAKSGM (MAKE THE $\underset{\approx}{\sigma}$ MATRIX)

MAKSGM makes the $\underset{\approx}{\sigma}$ matrix by taking conductivity information created by VEHICL and moving it to the right place in the appropriate matrix form (see 5.72.2) for ICCG.

MAKV (MAKE THE VOLTAGE VECTOR)

MAKV builds the voltage vector in the appropriate form (5.72.2) using a surface voltage list and the conductor voltage common block. The LAD number of the desired surface voltage list is passed as an argument to this routine.

MAKRHS (MAKE THE RIGHT HAND SIDE)

MAKRHS makes the right hand side of the vector equation solved by ICCG. The right hand side is equal to

$$\underset{\sim}{RHS} = \underset{\sim}{I}_t - \underset{\approx}{\sigma}\,\underset{\sim}{V} .$$

This is done by first calling MAKJ to calculate the current vector. Then MAKRHS finds the product of $\underset{\approx}{\sigma}\underset{\sim}{V}$ and subtracts it from the current. The right hand side of the vector equation is the same for both the explicit and implicit formulations so it only needs to be called once for the explicit step. Then the implicit step will just use the RHS found during the explicit calculation.

MAKJ (MAKE THE CURRENT VECTORS)

This routine finds the current vector $\underset{\sim}{I}$, in the appropriate form (see 5.72.2), given a list of surfaces voltages. To find the contributions to the current from the various sources, MAKJ calls the routines which calculate the incident, secondary, and backscattered electron fluxes and the incident and secondary ion currents.

Two lists are created by MAKJ, one is the total current, $\underset{\sim}{I}$. The other is the sum of the secondary currents, I_{sm}. The secondaries will be used in the future to locate surfaces which will need to be fixed.

MAKDJ (MAKE THE $d\underset{\sim}{I}/d\underset{\sim}{V}$ MATRIX)

MAKDJ is called before the solution of the implicit equation to create the $d\underset{\sim}{I}/d\underset{\sim}{V}$ matrix. To find $d\underset{\sim}{I}/d\underset{\sim}{V}$, MAKDJ uses

$$\frac{d\underset{\sim}{I}}{d\underset{\sim}{V}} = \frac{(I(t+\Delta t) - I(t)}{(V(t+\Delta t) - V(t)}$$

The surface voltages at $t+\Delta t$ were found during the explicit step. The surface currents at $t+\Delta t$ are found by calling MAKJ and giving it the voltages calculated by the explicit step.

To correct for the explicit charging equations tendency to overcharge, a limit is placed on the change in voltage. During the keyword input phase, the variable 'DVLIM' is set and this is the voltage change limit. Thus, for a surface i, $V_i(t+\Delta t)$ can be written

$$V_i(t+\Delta t) = V_i(t) + \Delta t V_i$$

where $|\Delta V_i|$ is the lesser of the explicity change or DVLIM. The limited voltage is then used to find the new surface current. The limited surface voltages at $t+\Delta t$ are also sent to MAKV to put in the appropriate form (see 5.73.2).

Once $\underset{\sim}{I}(t+\Delta t)$ and $\underset{\sim}{V}(t+\Delta t)$ have been calculated, dI/dV is calculated for each vector entry. If the value is not negative, $d\underset{\sim}{I}/d\underset{\sim}{V}$ is set equal to a zero for that vector entry. This is done to prevent the destabilizing effect of a positive current derivative in an element on the matrix diagonal (see 3.41). And since surface to surface interactions are not being dealt with presently, there are only diagonal elements in the $d\underset{\sim}{I}/d\underset{\sim}{V}$ matrix.

MATGEN (MATRIX GENERATION)

MATGEN finds the sum of the matrices which multiply the voltage vector, $\Delta\underset{\sim}{V}$, solved for by ICCG. It performs the sum

$$\underset{\sim}{M} = \frac{\underset{\sim}{C}}{\Delta t} + \underset{\sim}{\sigma} - \frac{d\underset{\sim}{I}}{d\underset{\sim}{J}}$$

SURVLT (SURFACE VOLTAGE)

SURVLT recovers from storage on file 19, the M matrix and RHS vector for use by ICCG. It also calls ICCG (4.32) which returns $\Delta\underset{\sim}{V}$. Currently, the initial guess for $\Delta\underset{\sim}{V}$ is $\Delta\underset{\sim}{V} = 0$. Also, we are presently fixing the main conductor, or spacecraft ground. The ground charges only according to its capacitance to the plasma which is small. All other surfaces and conductors have larger capacitance directly or indirectly to the ground (see Appendix A, and Section 4.51). Thus, fixing the ground simplifies the checkout of the charging algorithms. This restriction will be removed at the appropriate time.

SURSET (SURFACE VOLTAGE RESET)

SURSET uses the solution found by SURVLT to find the new surface and conductor voltages after charging.

5.82 GRAPHICAL CODE STRUCTURE

5.82.10 AN OVERVIEW OF SHONTL

SHONTL is a module designed to create the graphics commands for PLOTRD. It presents cross-sections of data generated by NTERAK as contour plots. Currently, axial slices of the positive ion density, the electric field potential, the sheath boundary, and the particle trajectories can be plotted.

The form of the plots can be varied by using keyword commands. The plot types and slice locations are also defined by keywords. The keywords and their use are described in Section 6.5.

5.82.11 SHONTL

This is the main routine of the module. SHONTL oversees the order of operations of the various phases of the plotting process (see Figure 5.82.11/1). It calls an initialization routine (PLINIT), an exit routine (PLEXIT) and cycles on the input routine and the plotting routine (PLINPT and GENPLT, respectively). EOF is a flag from PLINPT signalling the end of a plotting session.

5.82.12 PLINIT (PLOT INITIALIZATION)

PLINIT is the initialization routine. It opens POLAR data files 11 and 19, sets the raster graphics window, initializes common blocks, sets the defaults for plot descriptions and prints a welcoming message.

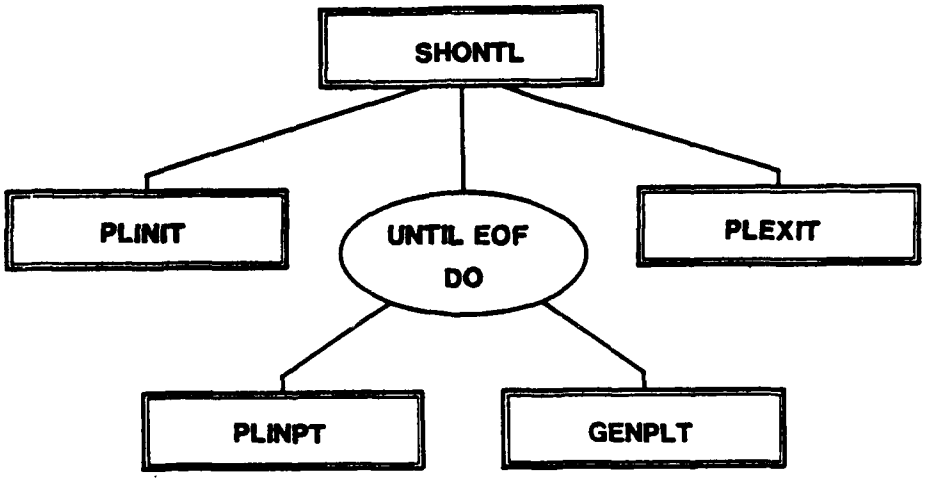


Figure 5.82.11/1. A general structure diagram of SHONTL.

5.82.13 PLINPT (PLOT INPUT)

This routine is the keyword input section of SHONTL. PLINPT recognizes the various keywords (described in detail in Section 6.5) and sets the appropriate flag for GENPLT. In addition to keywords for plotting features and to describe the data for plotting, keywords are also recognized which direct the data handling, control the SHONTL exiting and plotting procedure, and provide useful aids for the user. These include a 'HELP' command which provides a short manual for successful plot production, a 'DICT' command to call a routine to provide a complete dictionary of keywords and a 'WHAT' command to see which plotting features have been selected.

Additionally, soon there will be a 'PROMPT' command which will shift the input mode to a prompt input mode. This will be instructive the first few times SHONTL is used since it steps through the available features in sequence, describing each briefly.

All of the plotting features have default values set in PLINPT so that no harm will be done if a flag is not set in PLINPT (the default values are described in Section 6.5 also).

5.82.14 GENPLT (GENERATE PLOTS)

GENPLT calls the routines responsible for generating the pseudo-graphics calls used by PLOTRD. The flags set by PLINIT and PLINPT are used to decide which plotting routines to call.

To make a plot, GENPLT sets the constants it will need, reads the data to be plotted, generates the plot along with any extra features selected, and then prepares for the next plot.

5.82.15 PLEXIT (PLOT EXIT)

PLEXIT is the exit routine for SHONTL. It calls PLOTRD if plots were created and are to be printed.

REFERENCE, CHAPTER 5

5-1 "NASCAP Programmers' Reference Manual," S-CUBED Report SSS-R-82-5443 (DRAFT), March 1982.

6.0 OPERATING INSTRUCTIONS

In general, POLAR can be run in either an interactive or batch computing environment using keyword input. Commands controlling run characteristics, defining environmental constants, and choosing the amount of appropriate output are either recognized by the module's input routine or by a general input routine (POLINP) which processes non-module specific keywords such as the grid size and diagnostic keywords.

The exception to this general rule is the set of keywords used to define an object. These keywords are expected to be found within a definition file separate from the VEHICL runstream. Section 6.1 describes the intricacies of object definition in detail and includes several helpful examples.

The other sections describe the keywords pertinent to all of the modules (Section 6.7) or keywords appropriate to a particular module (Sections 6.2 to 6.5). Some keywords appear in several locations for the sake of convenience.

6.10 OBJECTS

POLAR objects are defined separately from the VEHICL runstream. VEHICL will look for the object definition file as unit 20 (CYBER) or as PREOBJ. on UNIVAC with PRE being a prefix given in the VEHICL runstream (Section 6.2).

POLAR objects are defined in an object grid of variable proportions subject to the limitation NX*NY*NZ = 9537; NY,NX $\leq$ 32; NZ $\leq$ 33. The keyin language is identical to NASCAP, except for the addition of slanted thin plates and the omission of booms (POLAR does not do booms), antennas, and thin triangular plates. If "empty space" and "object" both coexist in the same computational space what makes objects distinguishable? The answer is that POLAR can distinguish between volume elements that are filled (with object) and those that are empty (except for ambient plasma of course). Once we have this distinction it is easy to see how objects can be constructed by filling in collections of volume elements. For example, a simple cuboid may be constructed by filling in 2 x 3 x 4 = 24 elements as shown in Figure 6/1.

While arrangements of completely filled and completely empty cubes can be quite versatile in representing objects of many different shapes, more sophisticated representations are possible if we allow cubes to be partially filled (or partially empty). Only three partially filled cubes are allowed. These are shown in Figure 6/2.

While it is easy to see how objects might be constructed by filling or partially filling individual volume elements, a command structure that required the user to specify every element comprising an object would be very cumbersome to use. So several generalized objects are definable.

It is very important to note that POLAR objects must never touch the edge of the object grid defined by VEHICL (see Section 6.2).

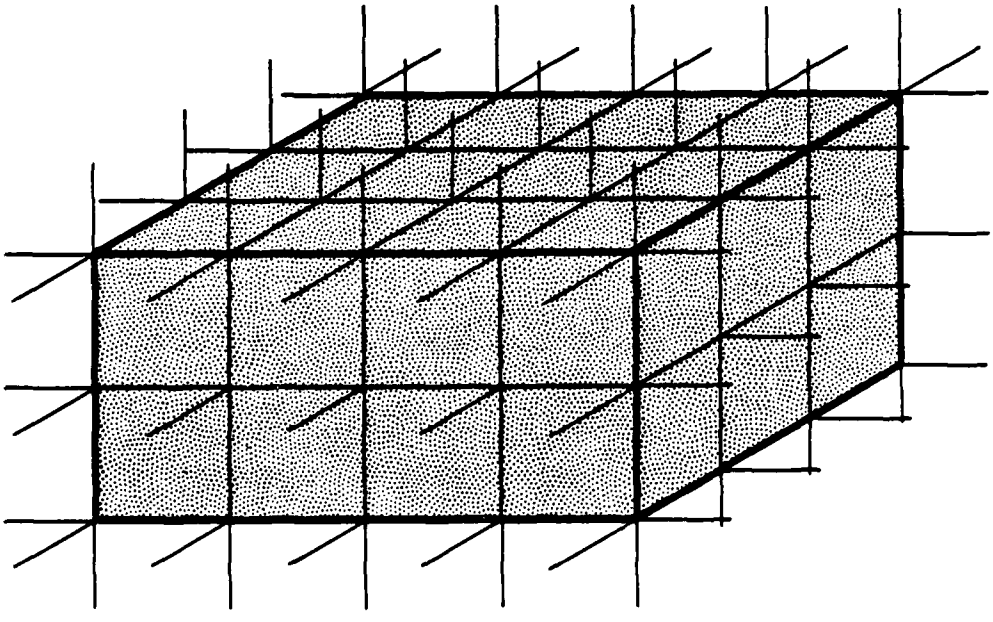


Figure 6/1. Cuboid made by filling in twenty-four volume elements.

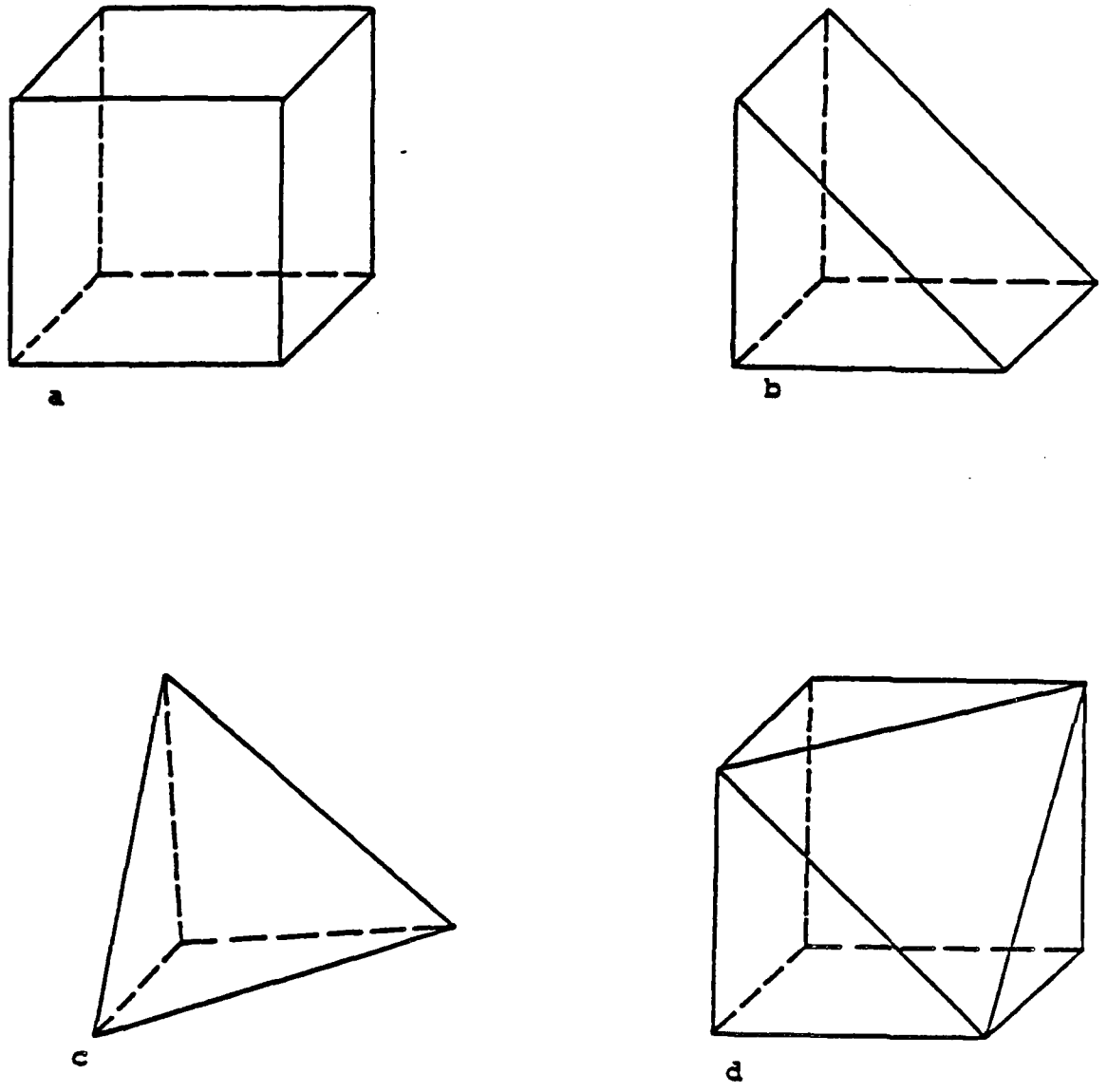


Figure 6/2. Four shapes of volume cells considered by the POLAR CODE: (a) empty cube; (b) wedge-shaped cell with 110 surface; (c) tetrahedron with 111 surface; (d) truncated cube with 111 surface.

6.10.10 BUILDING BLOCKS

To greatly simplify the user definition of objects, POLAR pre-defines commonly used shapes built up from individual elements. These shapes are called POLAR BUILDING BLOCKS. There are eight.

Flat Plate
Slanted Plate
Cuboid
Octagon
Quasisphere
Tetrahedron
Wedge
FIL111

These are shown in Figure 6/3. These basic shapes can be defined to be any size (within the inner grid). POLAR automatically includes the correct number of individual elements for the size of building block chosen by the user.

6.10.11 COMMANDS (OR HOW DO I ACTUALLY DEFINE AN OBJECT?)

The POLAR module VEHICL is responsible for recognizing and understanding the object defined by the user in the object definition file.

Each building block has its own keyword. For example, the quasi-sphere is associated with the word QSPHERE, and the cuboid (rectangular parallelepiped) with the word RECTAN. The building blocks and their keywords are summarized in Table 6/1.

Once VEHICL has read a building block keyword from the object definition file, it then expects to find several more lines (or cards) setting the block parameters. These might include the dimensions of

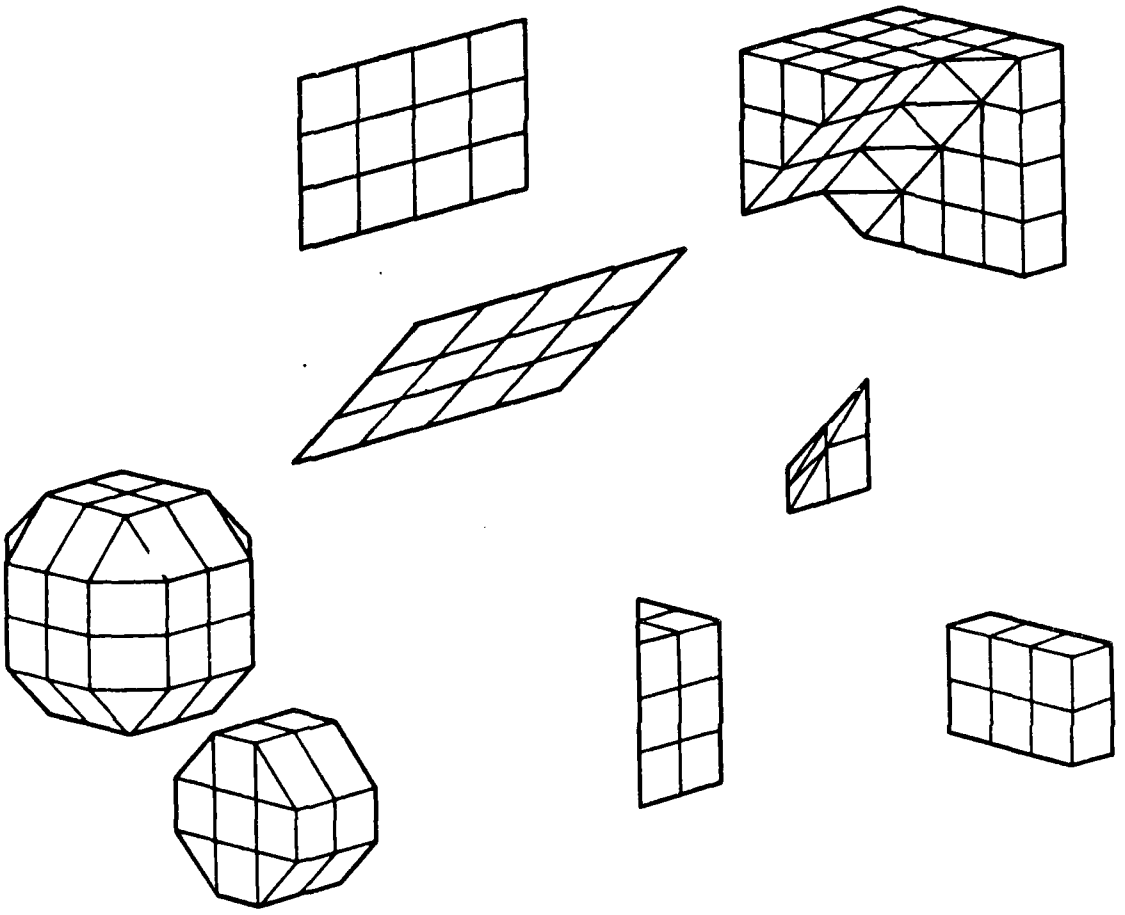


Figure 6/3. The eight building block types are shown here. The uppermost object shows a FIL111 smoothing a corner. Below, from left to right are quasi-sphere, octagon right cylinder, tetrahedron, wedge, and rectangular parallelepiped.

TABLE 6/1. POLAR BUILDING BLOCKS AND THEIR KEYWORDS

Keyword	Building Block Description
FIL111	Smooth inside of a diagonal corner
OCTAGON	Right octagonal cylinder
PATCHR	Surface of a rectangle
PATCHW	Diagonal face of a wedge
PLATE	Arbitrarily thin plate or cuboid
QSPHERE	Quasisphere
RECTAN	Cuboid or rectangular parallelepiped
SLANT	Thin plate slanted at 45°
TETRAH	Tetrahedron
WEDGE	Wedge derived from half a cube

the building block, its orientation and the materials that cover its surface. (Surface materials are discussed in Chapter 6.12.) Finally, VEHICL expects to find a line 'ENDOBJ' telling it that no more information referring to the present block is coming and to expect the next building block keyword. The information to be entered in the object definition file for each building block is summarized in Table 6/2. Note that numbers and words may be separated by one or more spaces on the same line. (Input is free-format.)

The keyword 'ENDSAT' signals the end of the satellite definition and should be included at the end of all vehicle descriptions.

6.10.12 PLATES AND PATCHES

A careful inspection of Table 6/1 will show that there are some building blocks that are not derived from cubic volume elements. These are the PLATE, SLANT, PATCHR and PATCHW.

PLATEs are arbitrarily thin cuboids (RECTANS). They are assumed to have only a top and a bottom, the sides being of negligible height. Flat plates always lie in one of the axis planes (XY, XZ, YZ). SLANTed plates lie along one axis, and at a 45° angle to the other two.

PATCHR and PATCHW are the surfaces only of a cuboid and wedge, respectively. They are used to change the surface material patterns of existing building blocks and should never be defined in spaces not already occupied by solid objects. (Objects defined to occupy the same space are explained in Section 6.14.)

TABLE 6/2. OBJECT DEFINITION - FILE 20

OBJECT DEFINITION SYNTAX	OBJECT DEFINITION EXAMPLES
RECTAN CORNER x y z DELTAS Δx Δy Δz (UP TO 6 SURFACE CARDS) ENDOBJ	RECTAN CORNER 3 −2 8 DELTAS 1 2 4 SURFACE +X ALUMINUM SURFACE −X ALUMINUM SURFACE +Y ALUMINUM SURFACE −Y ALUMINUM SURFACE +Z ALUMINUM SURFACE −Z ALUMINUM ENDOBJ
WEDGE CORNER x y z FACE materialname normal (type 110) LENGTH Δx Δy Δz (UP TO 4 SURFACE CARDS) ENDOBJ	WEDGE CORNER −3 2 1 FACE SIO2 −1 −1 0 LENGTH 1 1 3 SURFACE +X SIO2 SURFACE +Y SIO2 SURFACE +Z GOLD SURFACE −Z SIO2 ENDOBJ
TETRAH CORNER x y z FACE materialname normal (type 111) LENGTH Δx (UP TO 3 SURFACE CARDS) ENDOBJ	TETRAH CORNER −3 −2 8 FACE KAPTON 1 1 −1 LENGTH 2 SURFACE −X TEFLON SURFACE −Y KAPTON SURFACE +Z TEFLON ENDOBJ
OCTAGON AXIS x y z x' y' z' WIDTH w SIDE s (UP TO 3 SPECIAL SURFACE CARDS "+", "−", or "C") ENDOBJ	OCTAGON AXIS 3. 2. −6 3. 2. −9 WIDTH 3 SIDE 1 SURFACE + SILVER SURFACE − SILVER SURFACE C MAGNES ENDOBJ
OSPHERE CENTER x y z DIAMETER d SIDE s MATERIAL materialname ENDOBJ	OSPHERE CENTER 0. 0. 0 DIAMETER 4 SIDE 2 MATERIAL NPAINT ENDOBJ
SLANT CORNER x y z TOP materialnormal (type 110) BOTTOM material LENGTH Δx Δy Δz ENDOBJ	SLANT CORNER -1 -1 0 TOP TEFLON 1 1 0 BOTTOM KAPTON LENGTH 2 2 3 ENDOBJ

OBJECT DEFINITION SYNTAX	OBJECT DEFINITION EXAMPLES
FIL111 CORNERLINE x y z x' y' z' FACE materialname normal (type 111) ENDOBJ	FIL111 CORNER 3. 2 6. -5 4 6 FACE SOLAR −1 −1 −1 ENDOBJ
PLATE CORNER x y z DELTAS Δx Δy Δz TOP $\pm\binom{x}{y}$ materialname BOTTOM $\pm\binom{x}{z}$ materialname ENDOBJ	PLATE CORNER −1 −1 −10 DELTAS 2 2 0 TOP +Z CPAINT BOTTOM −Z CPAINT ENDOBJ
PATCHR CORNER x y z DELTAS Δx Δy Δz (UP TO 6 SURFACE CARDS) ENDOBJ	PATCHR CORNER 3 −2 8 DELTAS 1 0 1 SURFACE −Y SCREEN ENDOBJ
PATCHW CORNER x y z FACE materialname normal (type 110) LENGTH Δx Δy Δz (UP TO 4 SURFACE CARDS) ENDOBJ	PATCHW CORNER −3 2 7 FACE AQUADG −1 −1 0 LENGTH 1 1 1 ENDOBJ

NOTES "normal" is three values each either +1. 0. or −1

SURFACE CARD has the following format

SURFACE $\pm\binom{x}{y}$ materialname

SPECIAL SURFACE CARD is:

SURFACE $\left(\frac{\pm}{C}\right)$ materialname

OTHER OBJECT DEFINITION COMMANDS

ENDSAT	*Must be last card in file*
COMMENT	No effect
OFFSET i j k	Moves coordinate origin
CONDUCTOR n	Sets number of underlying conductor ($1 \leq n \leq 15$).
DELETE i j k	Deletes surfaces leaving empty cell
unrecognized word	Assumed to be name of new surface material. Next card scanned for parameters

6.10.13 SPECIAL SHAPES

FIL111 is a special shape designed to fill in "steps" whose corner line runs at 45° to the grid lines in any axis plane (i.e., XY, ZY, XZ) (Figure 6/4a). There are two kind of "steps" that can occur between POLAR building blocks. For example, a small cuboid on top of another creates four "steps" that lie along grid lines (Figure 6/4b). These may be "filled in" or smoothed by defining a WEDGE to lie along the corner line of the step. A second type of step is possible however when, for example, a tetrahedron or octagon is defined to sit on top of another building block. These steps have corner lines that run at 45° between grid lines. This is shown in Figure 6/4c. Such steps can be smoothed or filled in by a combination of tetrahedra and truncated cubes. This combination is supplied as the building block FIL111.

6.10.14 BUILDING BLOCK PARAMETERS (OR WHO'S ON NEXT?)

The very center of the object grid is assumed to be the origin of the coordinate system 0, 0, 0. Hence the grid itself extends from -8 to +8 in the X and Y directions and from -16 to +16 in the Z direction. This coordinate system is used to specify the position and size of the building blocks in the parameter "cards" or lines following the building block keyword. Let us examine the definition of each building block in detail to see how this works.

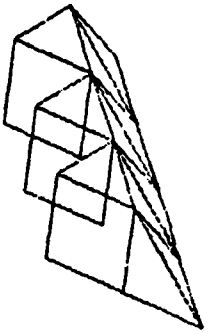


Figure 6/4a. A FIL111 building block all by itself.

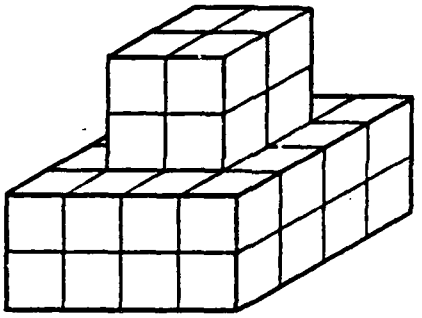


Figure 6/4b. "Steps" along grid lines.

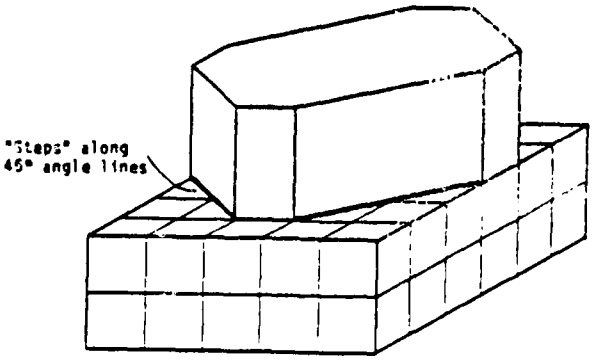


Figure 6/4c. "Steps" along 45° angle lines.

6.10.15 RECTAN

The following cards define a cuboid or rectangular parallelepiped:

```
RECTAN
CORNER   x y z
DELTAS   Δx, Δy, Δz
SURFACE  +X GOLD
SURFACE  -Y KAPTON
(Four more SURFACE cards for -X, +Y, +Z, -Z)
ENDOBJ
```

Notes:

1. RECTAN: is the building block keyword.

2. CORNER x y z: defines the coordinate of the lowest indexed corner of the cuboid (the one so that if you added up x + y + z it would give the lowest (least positive) number).

3. DELTAS Δx, Δy, Δz: gives the length of sides of the cuboid along the X, Y and Z axes. (Note that the edges of the cuboid must lie in the direction of the three axes.)

4. SURFACE +X GOLD: assigns the material GOLD (see Chapter 6.12) to the surface of the cuboid whose normal points in the +X direction. There are up to six surfaces that may be assigned materials (+X, -X, +Y, -Y, +Z, -Z). All surfaces that will eventually become a surface of the finished object (rather than become a connection to another building block) must be assigned a material. (For surfaces that are shared with other building blocks the material assigned is ignored.)

As an example, the following cards:

```
RECTAN
CORNER   -4  2  -1
DELTAS    3  2   5
SURFACE  +X  GOLD
SURFACE  +Y  GOLD
SURFACE  +Z  GOLD
SURFACE  -X  GOLD
SURFACE  -Y  GOLD
SURFACE  -Z  GOLD
ENDOBJ
```

define a gold bar extending from -4 to -1 in the X direction, 2 to 4 in the Y direction and -1 to +4 in the Z direction (Figure 6/5).

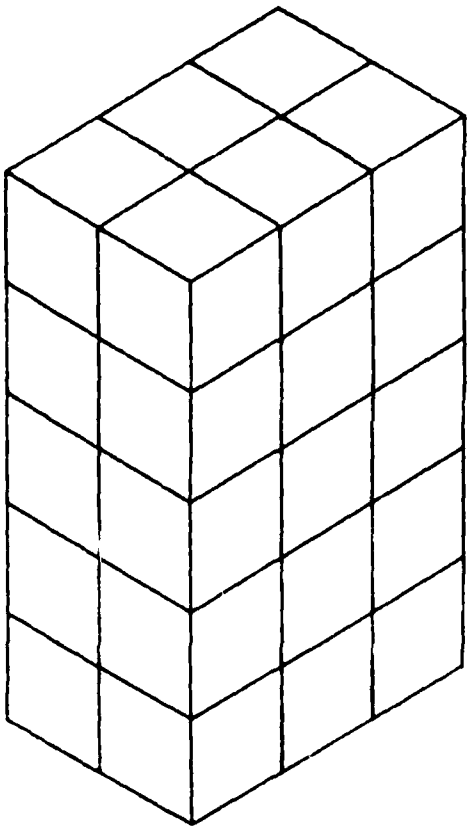


Figure 6.5. RECTAN.

6.10.16 PATCHR

PATCHR is defined in exactly the same way as RECTAN.

```
PATCHR
CORNER  x y z
DELTAS  Δx  Δy  Δz
<SURFACE card(s) (usually just one)>
e.g., SURFACE  +X  GOLD
ENDOBJ
```

PATCHR should only be defined within an existing object.(see 6.14).

6.10.17 WEDGE

The following cards define a right angled wedge:

```
WEDGE
CORNER   x  y  z
FACE     KAPTON  1 1 0
LENGTH   Δx  Δy  Δz
SURFACE  +X  TEFLON
<Up to four SURFACE cards>
ENDOBJ
```

Notes:

1. WEDGE: is the building block keyword.

2. CORNER x y z: defines the lowest indexed vertex of the right angled corner of the wedge (see note 2, 6.10.15).

3. FACE KAPTON 110: contains two pieces of information:

   a. 'KAPTON' assigns the material KAPTON to the surface of the face of the wedge. (The face is the sloping surface of the wedge.)

   b. '1 1 0' defines the direction of the normal to the face and hence the orientation of the wedge itself. The normal may point in any of the following directions only:

$\pm 1, \pm 1, 0$

$\pm 1, 0, \pm 1$

$0, \pm 1, \pm 1$

(For those of you not familiar with the '1 1 0' notation a '1 1 0' normal is a vector pointing to the coordinates X = 1, Y = 1 and Z = 0 from the origin.)

4. LENGTH Δx, Δy, Δz: gives the lengths of the sides of the wedge parallel to the X, Y and Z axes. To maintain symmetry two of these must be equal (i.e., the two right triangle sides).

5. SURFACE +X TEFLON: assigns the material 'TEFLON' (Section 6.12) to the surface whose normal points in the positive X direction. There are up to four remaining surfaces that may be assigned materials (see 6.10.15, note 4). These all have normals pointing along one of the axis directions. Along which axis direction they point depends on the orientation of the wedge or the choice of normal for the face (note 2). The possible combinations of face directions and remaining surface directions are summarized in Table 6/3. Cards defining materials for non-existent faces are ignored.

As an example, the following cards:

```
WEDGE
CORNER   0 0 0
FACE     GOLD  1 1 0
LENGTH   2 2 2
SURFACE  -X  GOLD
SURFACE  -Y  GOLD
SURFACE  +Z  GOLD
SURFACE  -Z  GOLD
ENDOBJ
```

define a wedge covered in gold with the origin as one of its corners and a face whose normal points between the X and Y axes in the XY plane. This is shown in Figure 6/6.

TABLE 6/3. DIRECTIONS OF SURFACE NORMALS ASSOCIATED WITH ALLOWED WEDGE ORIENTATION

Normal of WEDGE Face	Normals of Four Remaining Surfaces
1 1 0	-X, -Y, Z, -Z
-1 1 0	X, -Y, Z, -Z
1 -1 0	-X, Y, Z, -Z
-1 -1 0	X, Y, Z, -Z
1 0 1	-X, Y, -Y, -Z
-1 0 1	X, Y, -Y, -Z
1 0 -1	-X, Y, -Y, Z
-1 0 -1	X, Y, -Y, Z
0 1 1	X, -X, -Y, -Z
0 -1 1	X, -X, Y, -Z
0 1 -1	X, -X, -Y, Z
0 -1 -1	X, -X, -Y, -Z

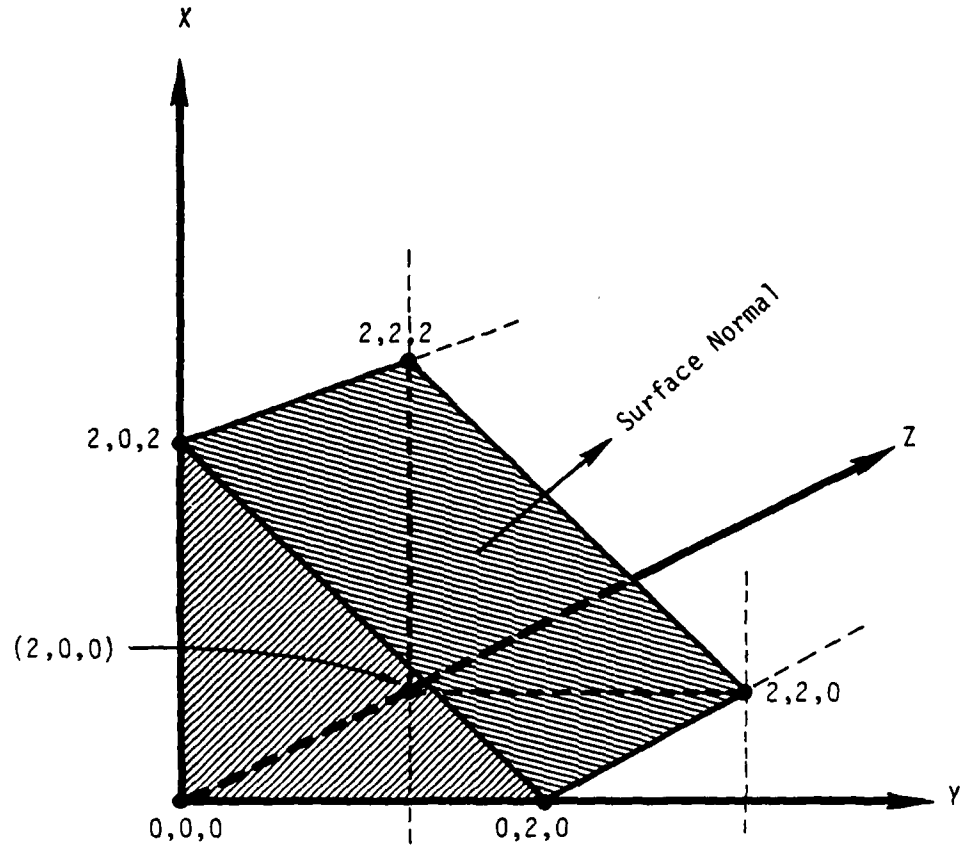


Figure 6/6. Wedge defined with surface normal 110 and corner 0,0,0.

6.10.18 PATCHW

PATCHW is defined in exactly the same way as a wedge.

```
PATCHW
CORNER  x y z
FACE  GOLD  1  -1  0
LENGTH  Δx  Δy  Δz
<Up to four SURFACE cards (usually just one)>
ENDOBJ
```

Like PATCHR (6.10.16) it may only be used to define a wedge inside another building block. This is explained further in Section 6.14.

6.10.19 TETRAH

The following cards define a tetrahedron:

```
TETRAH
CORNER   x  y  z
FACE  ALUMINUM  1  1  -1
LENGTH   Δx
SURFACE  -X  TEFLON
SURFACE  -Y  TEFLON
SURFACE  +Z  TEFLON
ENDOBJ
```

Notes:

1. TETRAH is the building block keyword.

2. CORNER x y z: defines the coordinates of the right angled corner of the tetrahedron. There is only one of these. (It corresponds to the corner of the partially filled cubic volume element that is actually filled.)

3. FACE 1 1 -1: assigns the material ALUMINUM to the unique face of the tetrahedron opposite the right angled corner. '1 1 -1' gives the direction of this face's surface normal and hence the orientation of the tetrahedron. The following directions only are allowed:

   $\pm 1, \pm 1, \pm 1$

   (This notation is the same as explained in 6.10.17, note 3.)

4. LENGTH Δx: gives the length of the sides along the X, Y and Z axis directions. (These must all be equal to preserve symmetry.)

5. SURFACE -X TEFLON: assigns the material teflon to the remaining surface with surface normal pointing along the negative X axis direction. Up to three surfaces remain to be assigned materials (see 6.10.15, note 2). The surface normals of these surfaces depend on the orientation of the tetrahedron and hence the normal of the "face". Table 6/4 summarizes these relationships. Definitions of non-existent surfaces are ignored.

TABLE 6/4. DIRECTIONS OF SURFACE NORMALS ASSOCIATED WITH ALLOWED TETRAHEDRON ORIENTATIONS

Normal of TETRAHedron Face	Normals of Three Remaining Surfaces
1 1 1	-X, -Y, -Z
-1 1 1	X, -Y, -Z
1 -1 1	-X, Y, -Z
1 1 -1	-X, -Y, Z
-1 -1 1	X, Y, -Z
-1 1 -1	X, -Y, Z
1 -1 -1	-X, Y, Z
-1 -1 -1	X, Y, Z

As an example, the following cards:

```
TETRAH
CORNER  0  0  0
FACE  KAPTON  1  1  1
LENGTH  2
SURFACE  KAPTON  -X
SURFACE  KAPTON  -Y
SURFACE  KAPTON  -Z
ENDOBJ
```

define a tetrahedron with its right angle corner at the origin and the normal of the opposite face pointing between the positive X, Y and Z axes. This is shown in Figure 6/7.

6.10.20 OCTAGON

The following cards define a right octagonal cylinder:

```
OCTAGON
AXIS  x y z   x' y' z'
WIDTH  w
SIDE  s
SURFACE  + GOLD
SURFACE  - GOLD
SURFACE  C GOLD
ENDOBJ
```

Notes:

1. OCTAGON: is the building block keyword.

2. AXIS x y z x' y' z': defines both the direction of the symmetry axis and the height of the cylinder. The symmetry axis must be parallel to one of the axis directions. Thus two of the coordinate pairs (x, x'), (y, y') and (z, z') must be identical. For example,

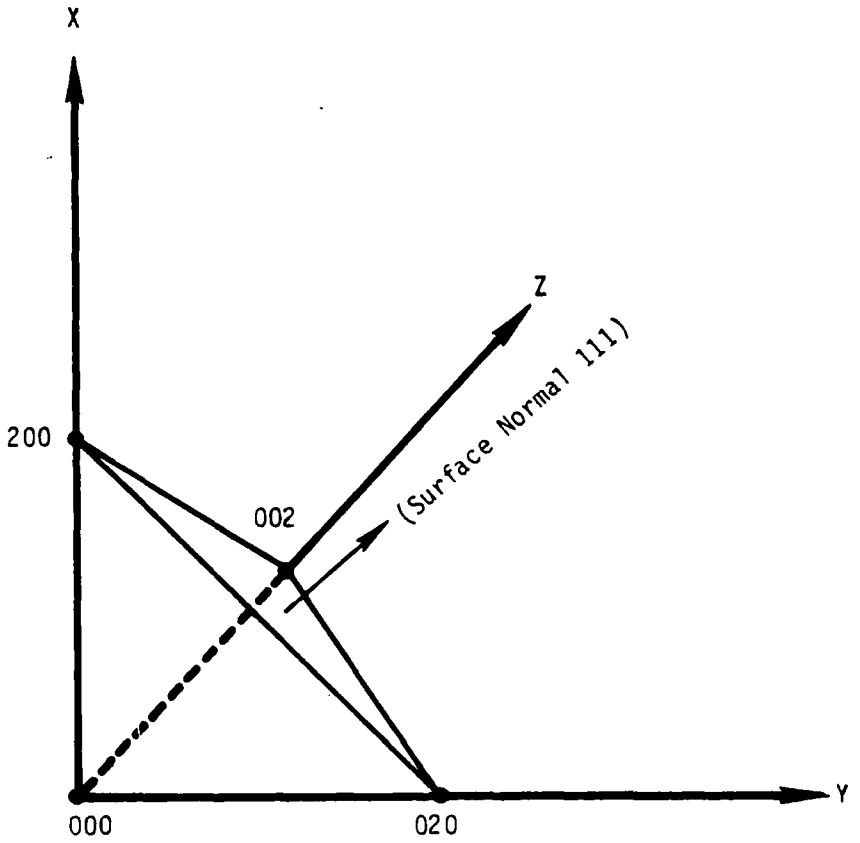


Figure 6/7. Tetrahedron defined with its "corner" at 000 and a surface normal 111.

'AXIS 6 3 -2 7 4 -2'

would define an axis that was not parallel to the X, Y or Z directions. However,

'AXIS 6 3 -2 7 3 -2'

defines an axis parallel to the X direction and is acceptable.

The height of the cylinder is given by the difference in coordinates along the axis direction. (For example, in the case above, the axis is one mesh unit long.)

3. WIDTH w: gives the width of the octagonal cross-section of the cylinder as w. If WIDTH is chosen to be odd, the axis must be moved or the sides of the cylinder will lie halfway across a volume element. POLAR automatically moves the axis +1/2 a mesh unit in each direction in the plane perpendicular to it.

4. SIDE s: gives the length of one of the sides of the octagonal cross-section that lies in an axis direction. The symmetry relationship between the width and the ides of the cross-section is shown in Figure 6/8. To maintain this relationship the side must always be an even number of mesh units less than the width. This means that they both either must be odd or both even numbers of mesh units.

5. SURFACE + GOLD: assigns the material GOLD to the top surface of the cylinder. '-' and 'C' replacing the '+' assign surface materials to the bottom or side cylindrical surface, respectively. Only those surfaces that will eventually become surfaces of the completed object need be assigned a material.

As an example, the following cards:

```
OCTAGON
AXIS  2  -4  6    2  -4  10
WIDTH  5
SIDE  3
SURFACE + TEFLON
SURFACE - TEFLON
SURFACE C TEFLON
ENDOBJ
```

defines a right octagonal cylinder covered in teflon. The symmetry axis lies along the Z direction and the height of the cylinder is four mesh units. Because the WIDTH is odd the axis is imagined to pass through the point 2 1/2, -3 1/2 in the X Y plane. Hence the top and

bottom faces run from X = 0 to X = 5 and from Y = -6 to Y = -1. The coordinates of the top of the cylinder are shown in Figure 6/8. A three-dimensional view is shown in Figure 6.9.

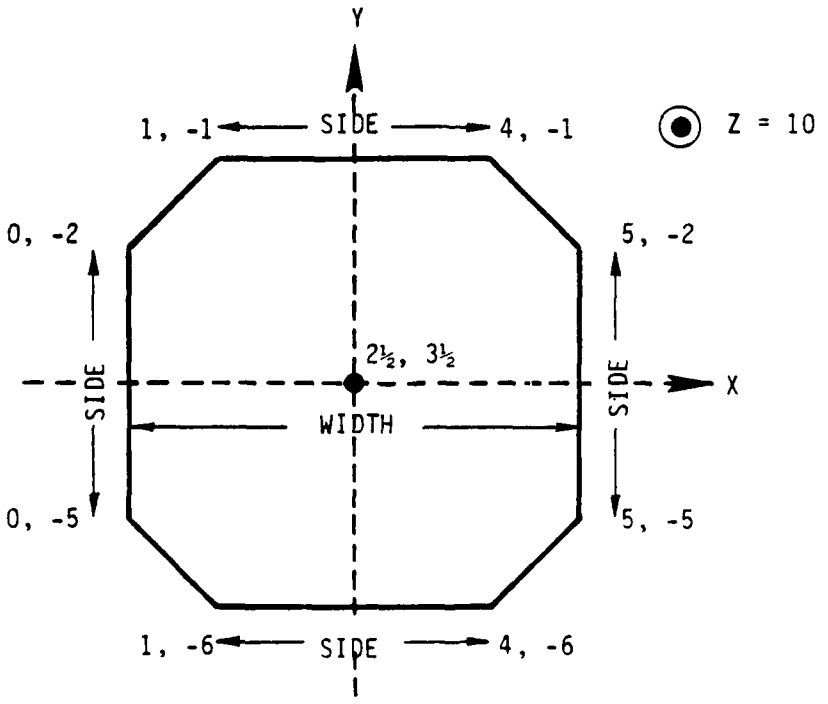


Figure 6/8. Top of an OCTAGON.

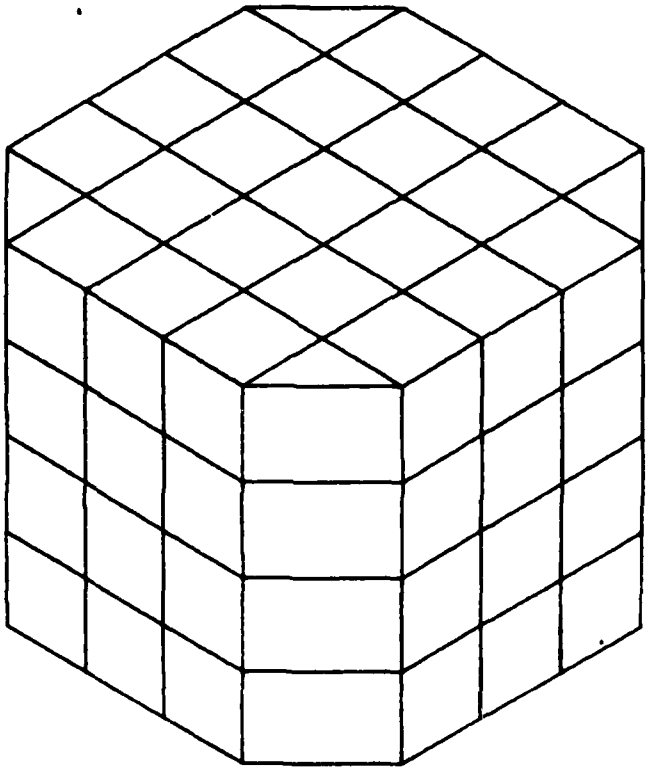
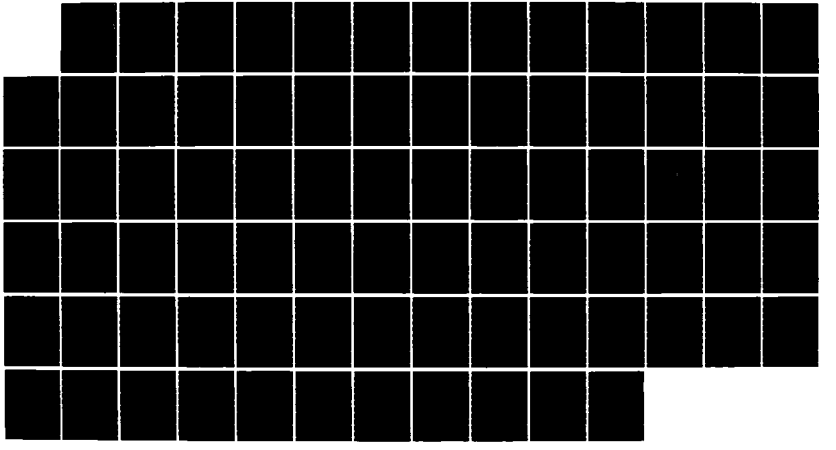
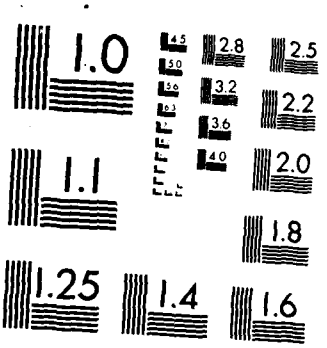


Figure 6/9. Octagon.

AD-A173 758 POLAR USER'S MANUAL(U) S-CUBED LA JOLLA CA 3/3
J R LILLEY ET AL. OCT 85 SSS-R-86-7563 AFGL-TR-85-0246
F19628-82-C-0081
UNCLASSIFIED F/G 22/1 NL





6.10.21 QSPHERE

The following cards define a quasisphere:

```
QSPHERE
CENTER  x  y  z
DIAMETER  d
SIDE  s
MATERIAL  SiO2
ENDOBJ
```

Notes:

1. QSPHERE: is the building block keyword.

2. CENTER x y z: defines the center of the sphere to be at coordinates X, Y, Z.

3. DIAMETER d: defines the diameter of the sphere to be d mesh units. The quasisphere can be thought of as an octagonal cross-section (like the top of an OCTAGON (see 6.10.20)) rotated about an axis in the cross-section plane. The diameter then corresponds to the WIDTH for a two-dimensional octagonal section. The same restrictions then apply: An odd value for the DIAMETER causes POLAR to automatically move the CENTER by +1/2 a mesh unit in the X, Y and Z directions.

4. SIDE s: sets the length of a side lying in one of the axis planes (e.g., X Y plane). Like the OCTAGON, the SIDE and DIAMETER must differ by an even number of mesh units.

5. MATERIAL SIO2: assigns the material SIO2 to the whole sphere surface.

As an example, the following cards:

```
QSPHERE
CENTER  1  -3  5
DIAMETER  7
SIDE  3
MATERIAL  SILVER
ENDOBJ
```

define a silver sphere centered at 1 1/2, -2 1/2 and 5 1/2. The sphere extends along the axis direction as follows:

x from -2 to 5
y from -6 to 1
z from 2 to 9

(See Figure 6/10.)

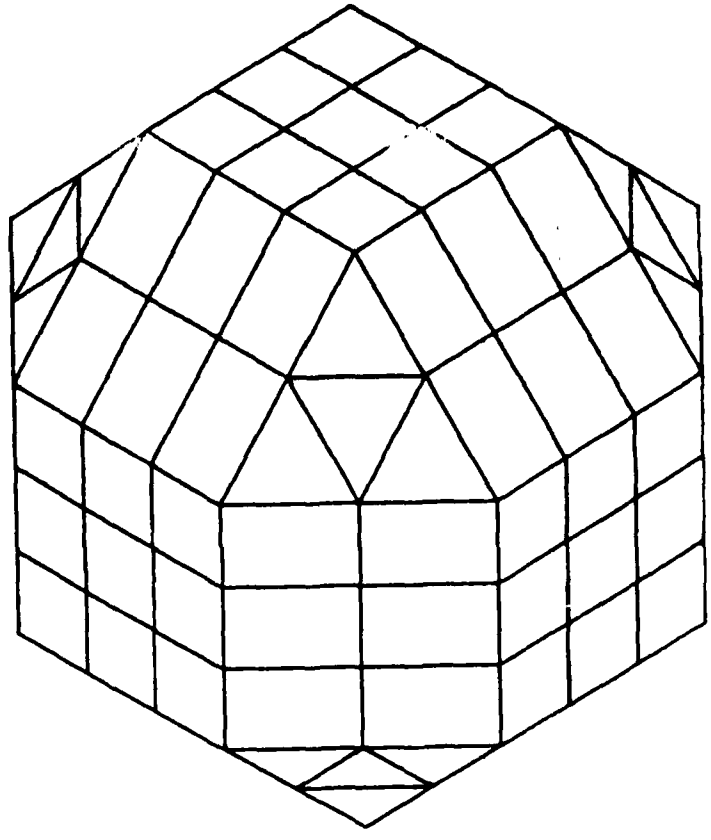


Figure 6/10. QSPHERE.

6.10.22 FIL111

The following cards define a FIL111:

```
FIL111
CORNERLINE  x  y  z   x'  y'  z'
FACE  KAPTON  1  -1  -1
ENDOBJ
```

Notes:

1. FIL111: is the building block keyword.

2. CORNERLINE x y z x' y' z': defines both the length and the direction of the "step" FIL111 is to fill. The line must lie in one of the axis planes (XY, XZ, YZ) and must have a direction lying 45° to two of the axes. This means that one pair of the coordinates (x', x), (y, y') (z, z') must be identical and the other two pairs must differ by the same magnitude. For example,

   'CORNERLINE 1 2 3 4 5 6'

   is unacceptable since all three coordinate pairs change. The following correct example

   'CORNERLINE 1 2 3 -1 4 3'

   defines a line in the XY plane (Z is constant) with $\Delta x = -2$, and $\Delta y = +2$. Hence the line is $2\sqrt{2}$ units in length and runs at 45° between the positive Y axis and the negative X axis.

3. FACE KAPTON 1 -1 -1: assigns the material KAPTON to the exposed surfaces of the FIL111 and defines its orientation via the surface normal of its exposed face: 1 -1 -1. The surface normal can only be combinations of

   $\pm 1 \quad \pm 1 \quad \pm 1$

   Only certain choices of corner line direction are consistent with each choice of FACE normal. If we subtract the x y z, x' y' z' coordinates defined in corner line

   $\Delta x = x' - x$
   $\Delta y = y' - y$
   $\Delta z = z' - z$

   then the surface normal $n_1\ n_2\ n_3$ (e.g., 1 1 1) must be orthogonal to Δx, Δy, Δz, i.e.,

   $$\Delta x \cdot n_1 + \Delta y \cdot n_2 + \Delta z \cdot n_3 = 0.$$

With the choice 1 2 3 -1 4 3 for the corner line coordinates only 1 1 $\pm$1 or -1 -1 $\pm$1 faces are permissible, e.g.,

-2. -1 + 2. -1 + 0. +1 = 0.

However, with -1 +1 $\pm$1

-2. -1 + 2.1 + 0. +1 = 4

the vectors are not orthogonal and so are not allowed.

As an example, the following cards:

```
FIL111
CORNERLINE  1  4  -6   1  7  -3
FACE  GOLD  -1  1  -1
ENDOBJ
```

defines a FIL111 covered with gold smoothing a step with a corner line running from 1 4 -6 in the YZ plane, between the positive Y and Z axis to 1 7 -3. The face of the FIL111 points in the negative X and Z directions and positive Y direction. (See Figure 6/11.)

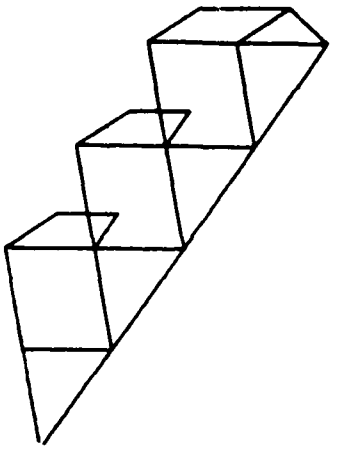


Figure 6/11. FIL111.

6.10.23 PLATE

The following cards define a PLATE:

PLATE
CORNER x y z
DELTAS Δx Δy Δz
TOP $\pm\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$ ALUMIN
BOTTOM $\pm\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$ KAPTON
ENDOBJ

Notes:

1. PLATE: is the building block keyword.

2. CORNER x y z: defines the vertex of the thin plate with the lowest indices (see 6.10.15, note 2).

3. DELTAS Δx Δy Δz: defines the length of the plate along the three axis directions. A PLATE may be thought of as a cuboid (or RECTAN) (see 6.10.15) with zero thickness in one direction. Hence one of Δx, Δy and Δz must be zero. For example, if Δy is chosen to be zero the PLATE will lie in the xz plane.

4. TOP $\pm\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$ ALUMIN:
assigns the material ALUMIN to the TOP surface of the plate. The "TOP" surface may be either in a + or - axis direction. This choice is arbitrary unless a "double point" conflict is possible. Double point conflicts are explained in Section 6.11.11.

5. BOTTOM $\pm\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$ KAPTON:
assigns the material KAPTON to the other side of the plate. If "top" were chosen as +X then bottom must be -X, and so on. Note that the choice of x, y or z must coincide with the Δx, Δy or Δz chosen to be zero.

As an example, the cards

```
PLATE
CORNER  0  0  0
DELTAS  0  2  2
TOP  -X  TEFLON
BOTTOM  +X  GOLD
ENDOBJ
```

defines a 2 x 2 thin plate with gold on the +X side and teflon on the -X side lying in the YZ plane. (See Figure 6/12.)

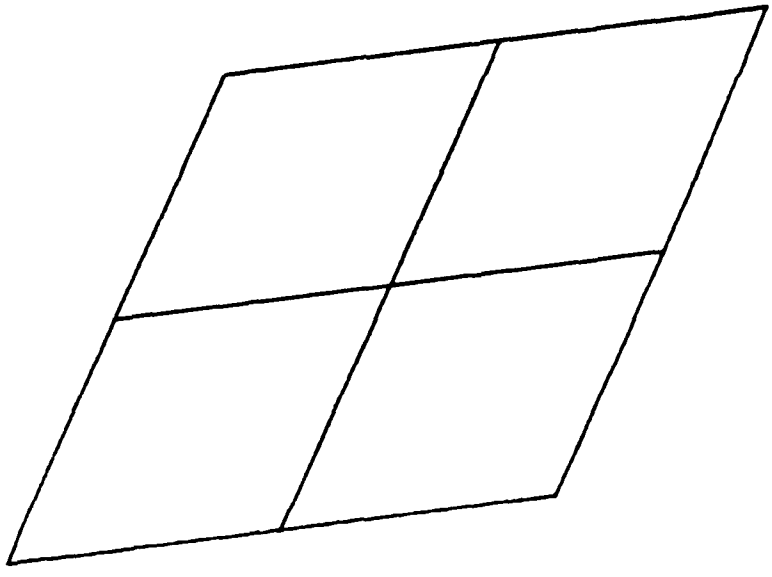


Figure 6/12. PLATE.

6.10.24 SLANT

The SLANT object should be thought of as a WEDGE with the 100-type surfaces left undefined. The FACE card is transformed to a TOP card, and a BOTTOM card is added. The CORNER is remote from the slanted plate, and is where the CORNER of a WEDGE would be if we were defining the FACE of a WEDGE. Similarly, the LENGTH card corresponds to that of a WEDGE. The syntax is

```
SLANT
CORNER  ix  jy  kz
TOP     matl  nx  ny  nz
BOTTOM  matl
LENGTH  lx  ly  lz
ENDOBJ
```

6.10.25 MORE OBJECT DEFINITION KEYWORDS

In addition to the building block keywords and the parameter cards that follow VEHICL also recognizes a few other keywords. With these and the building blocks a complete object definition file can finally be constructed. Let us examine the remaining keywords and their effect one by one.

ENDSAT

Just as ENDOBJ terminates a set of building block parameter cards, so the keyword 'ENDSAT' terminates the whole object definition file. After reading an 'ENDSAT' card VEHICL stops trying to read any more keywords from the object definition file and begins to process the information it has. Note that ALL object definition files must end with an 'ENDSAT' card.

COMMENT

VEHICL ignores anything written on the same 80 character line (or card) that begins with the keyword 'COMMENT'. This allows the user to include notes or reminders in long and complicated object definition files, e.g.,

```
.
.
.
COMMENT  DEFINE OCTAGONAL BODY
OCTAGON
AXIS  -6  0  2    -8  0  2
.
.
.
```

OFFSET

The card

OFFSET x y z

relabels the grid center to be (NX+1)/2-X, (NY+1)/2-Y, (NZ+1)/2-Z, 9-x, 9-y, 17-z for a 17 x 17 x 35 inner grid. The so-called "absolute" POLAR coordinate system labels the axes from 1 to 17 in the X and Y direction and 1 to 33 in the Z direction. In this "absolute system" the center of the grid, which we had previously labeled (0, 0, 0,) becomes (9, 9, 17). So we may move from the more intuitive (0, 0, 0) centered system to the "absolute" system with the command

OFFSET 0 0 0

To illustrate the difference the following two sets of cards define a sphere centered at the center of the grid.

Default coordinate system	Absolute coordinate system
QSPHERE	OFFSET 0 0 0
CENTER 0 0 0	QSPHERE
DIAMETER 7	CENTER 9 9 17
SIDE 3	DIAMETER 7
MATERIAL KAPTON	SIDE 3
ENDOBJ	MATERIAL KAPTON
	ENDOBJ

The coordinate system may be adjusted with the OFFSET command anytime at any point between building blocks. The active system is always the default or the coordinate system defined by the most recent OFFSET.

CONDUCTOR

POLAR allows for both insulating and conducting materials (Chapter 6.12). It assumes that all surface materials cover an underlying conductor. Up to 15 separate conductors are allowed. Each building block is associated with a particular conductor. This association is made by preceding all building block definitions associated with the first conductor with the card:

CONDUCTOR 1

Similarly, blocks associated with a second conductor are preceded by the card

CONDUCTOR 2

and so on. If no CONDUCTOR card is included in the object definition file all building blocks will be associated with CONDUCTOR 1. In the same way any building blocks defined before VEHICL encounters a card

CONDUCTOR n (n > 1)

will be associated with conductor 1. All subsequent blocks will be associated with conductor n, until another conductor card is encountered.

It is conventional to choose conductor 1 as the satellite ground conductor. Skipping conductor numbers is not recommended.

DELETE

DELETE allows the user to modify building blocks already defined by selectively "deleting" filled or partially filled cells (i.e., "deleting" them by making them empty).

DELETE x y z

empties the filled cell with the indices of its lowest index vertex given by x y z. (The lowest index vertex is the one with the sum of its X, Y and Z coordinates equal to the least positive number.) The coordinates x, y, z refer to the coordinate system presently active (i.e., the default system or that associated with the most recent OFFSET command). The DELETE command requires great care in its use. It does not assign materials to surfaces that are newly exposed by the removal of a filled element. The user must do this by defining a new object or objects with surfaces that coincide with those newly exposed. This is most easily done by overlaying objects (6.14).

COMPRESS

Since there is a limit of 1250 to the number of surfaces on an object (the rectangle defined in Figure 6/5 has 62 surfaces), large complex objects sometimes contain interior surfaces which need to be removed. Normally these surfaces are removed when the satellite definition is complete or when the number of surfaces exceeds the surface limit between building blocks, COMPRESS forces the existing interior surfaces to be removed immediately. An example of the syntax is

```
    .
    .
ENDOBJ
COMPRESS
RECTAN
    .
    .
```

MARKTB

Sometimes when defining odd objects with unusual geometries, undefined double points will be found. One way to patch the object is to use the MARKTB command to define an element and its associated surfaces to be TOPs or BOTTOMs.

```
MARKTB  1  2  2  TOP
MARKTB  1  2  1  BOTTOM
```

The first MARKTB command forces the cell with lowest index vertices of (1,2,2) to be marked as a TOP element. The second line marks the element below the first in the Z-direction as a BOTTOM cell. It should be noted that these index vertices will be affected by previous OFFSET commands. (See Section 6.11.11 for more hints for dealing with double points.)

OTHER WORDS

Any other words that VEHICL reads in the object definition file are assumed to be the names of new materials and VEHICL then expects three more cards defining the material properties (6.12) to follow immediately.

6.11 DEFINING AN OBJECT: AN EXAMPLE

The input file of Figure 6/13 defines an object consisting of an ALUMINUM slab, trimmed with four KAPTON wedges and four TEFLON tetrahedra, and topped with a GOLD sphere. Three views of the resulting object are shown in Figure 6/14.

```
1.    COMMENT  ALUMINUM SLAB                          ┐
2.        RECTAN                                      │
3.            CORNER -3 -4 -1                         │
4.            CELTAS 6 8 1                            │ CUBOID
5.            SURFACE +Z ALUMINUM                     │
6.            SURFACE -Z ALUMINUM                     │
7.            ENDOBJ                                  ┘
8.    COMMENT  FOUR KAPTON WEDGES                     ┐
9.        WEDGE                                       │
10.           CORNER -3 -4 -1                         │
11.           FACE KAPTON -1 0 1                      │
12.           LENGTH 1 8 1                            │
13.           SURFACE -Z KAPTON                       │
14.           ENDOBJ                                  │
15.       WEDGE                                       │
16.           CORNER -3 -4 -1                         │
17.           FACE KAPTON 0 -1 1                      │
18.           LENGTH 6 1 1                            │ FOUR WEDGES
19.           SURFACE -Z KAPTON                       │
20.           ENDOBJ                                  │
21.       WEDGE                                       │
22.           CORNER 3 -4 -1                          │
23.           FACE KAPTON 1 0 1                       │
24.           LENGTH 1 8 1                            │
25.           SURFACE -Z KAPTON                       │
26.           ENDOBJ                                  │
27.       WEDGE                                       │
28.           CORNER -3 4 -1                          │
29.           FACE KAPTON 0 1 1                       │
30.           LENGTH 6 1 1                            │
31.           SURFACE -Z KAPTON                       │
32.           ENDOBJ                                  ┘
33.   COMMENT  FOUR TEFLON TETRAHEDRA                 ┐
34.       TETRAHEDRON                                 │
35.           CORNER -3 -4 -1                         │
36.           FACE TEFLON -1 -1 1                     │
37.           LENGTH   1                              │
38.           SURFACE -Z TEFLON                       │
39.           ENDOBJ                                  │
40.       TETRAHEDRON                                 │
41.           CORNER 3 -4 -1                          │
42.           FACE TEFLON 1 -1 1                      │
43.           LENGTH   1                              │ FOUR
44.           SURFACE -Z TEFLON                       │ TETRAHEDRA
45.           ENDOBJ                                  │
46.       TETRAHEDRON                                 │
47.           CORNER 3 4 -1                           │
48.           FACE TEFLON 1 1 1                       │
49.           LENGTH   1                              │
50.           SURFACE -Z TEFLON                       │
51.           ENDOBJ                                  │
52.       TETRAHEDRON                                 │
53.           CORNER -3 4 -1                          │
54.           FACE TEFLON -1 1 1                      │
55.           LENGTH   1                              │
56.           SURFACE -Z TEFLON                       │
57.           ENDOBJ                                  ┘
58.   COMMENT  ALL TOPPED BY A GOLD SPHERE            ┐
59.       SSPHERE                                     │
60.           CENTER 0 0 2                            │
61.           DIAMETER 4                              │ SPHERE
62.           SIDE 2                                  │
63.           MATERIAL GOLD                           │
64.           ENDOBJ                                  ┘
65.       ENDSAT
```

Figure 6/13. Object definition example.

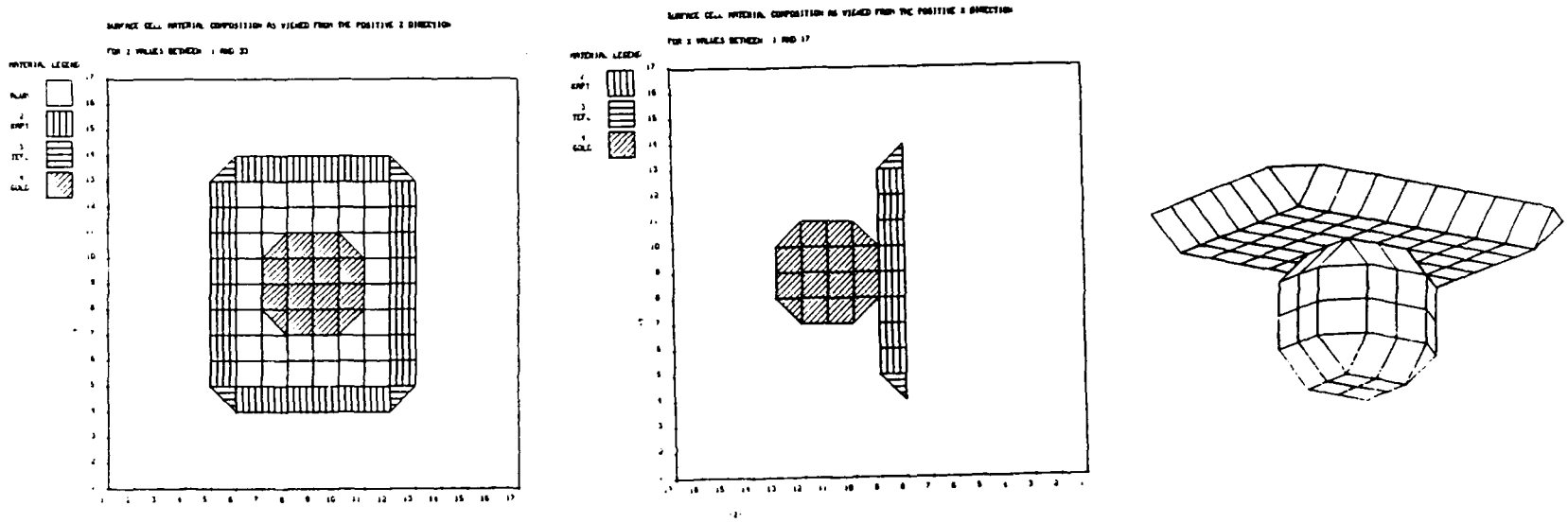


Figure 6/14. Three views of object defined by input of Figure 6/13.

6.11.10 LIMITATIONS IN OBJECT DEFINITION

It is probably fair to say that you can link building blocks together and nine times out of ten there will not be a problem. This section deals with the other one time out of ten, when what appears to be a perfectly reasonable combination of building blocks is rejected by VEHICL. We itemize here a rather formidable list of object definition "don'ts". However, you should remember that it takes hard work to break more than one or two of these rules defining any one object if you use a little common sense.

1. All exposed surfaces must be assigned materials.

2. The parameter cards for each building block, discussed in Section 6.10.14, must appear in the order shown, and no other.

3. The object must not touch the object grid boundary planes at any point.

4. Thin plates sharing the same volume element can do so only if the TOP face of one shares volume with the TOP of the other, or the BOTTOM face of one shares volume with the BOTTOM face of the other. TOP faces may not share volume elements with BOTTOM faces.

5. Thin plates may only intersect each other at the edges or corners.

6. Double points must be assigned TOP and BOTTOM sets (see Section 6.11.11).

(Rules 4 through 6 are all manifestations of conflicts involving double and triple points.)

6.11.11 DOUBLE POINTS

Thin plates may have different potentials on their two surfaces, yet they occupy only one plane of grid points. These grid points must therefore be associated with two distinct sets of potentials. For this reason they are called <u>double points</u>. The two sets of potentials associated with each half of the double points are distinguished by calling one set 'TOP' and one set 'BOTTOM'. Recall (6.10.23) that the

surfaces of a thin plate may be defined as 'TOP' or 'BOTTOM' regardless of whether their surface normal points along a positive or negative axis direction: The TOP and BOTTOM definition refers to the (arbitrary) choice of which set of potentials (TOP or BOTTOM) to associate with each surface. When double points share a volume element they must all be of the same type; i.e., all TOP or all BOTTOM. This is the basis for rule 4 in Section 6.11.10.

Double points also occur when other building blocks touch in such a way that their single points come together to form a common vertex of two "disjoint" volume elements. By "disjoint" volume elements we mean elements physically separated from each other by solid surfaces. This is shown for two cuboids touching along one edge only in Figure 6/15. The row of points along the touching edges are double points and one set must be defined as BOTTOM. This may be done by defining a thin plate touching the common edge. If the exterior surface of the plate pointing into one of the disjoint volumes is 'BOTTOM' then the half of the double point associated with the other disjoint volume becomes 'TOP'.

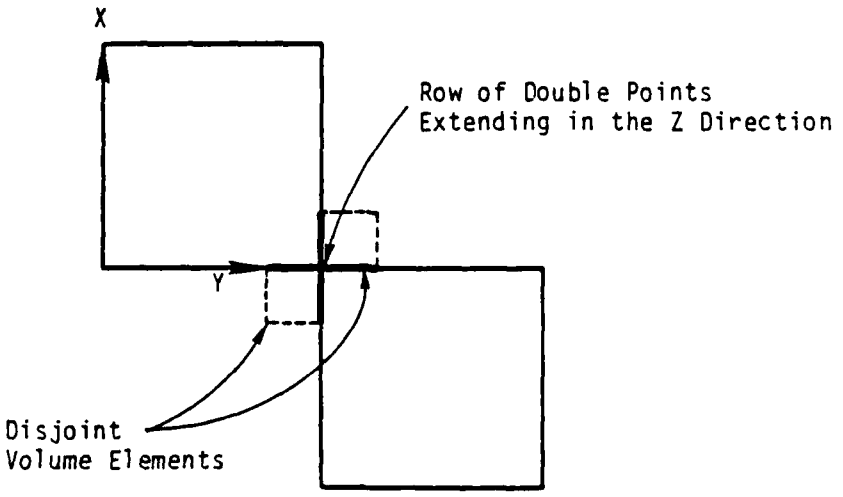


Figure 6/15. Profile of two cuboids sharing a common edge and resultant double points. Heavy lines show possible orientations for the definition of a thin plate to resolve the conflict.

Because of the way surface cell potentials are assigned to grid points, the edges of thin plates are only single points. However, a thin plate touching another building block with its edge creates a row of double points similar to that caused by two cuboids touching at an edge (Figure 6/16). These double points are automatically assigned TOP and BOTTOM sets.

Double point ambiguities can also be resolved on an element by element basis using the MARKTB command discussed in Section 6.10.25.

6.11.12 TRIPLE POINTS

A triple point is said to occur when a vertex is common to three or more disjoint volume elements. Triple points are illegal! The easiest way to get a triple point is to define one thin plate passing through another. This is not allowed (rule 5, Section 6.11.10).

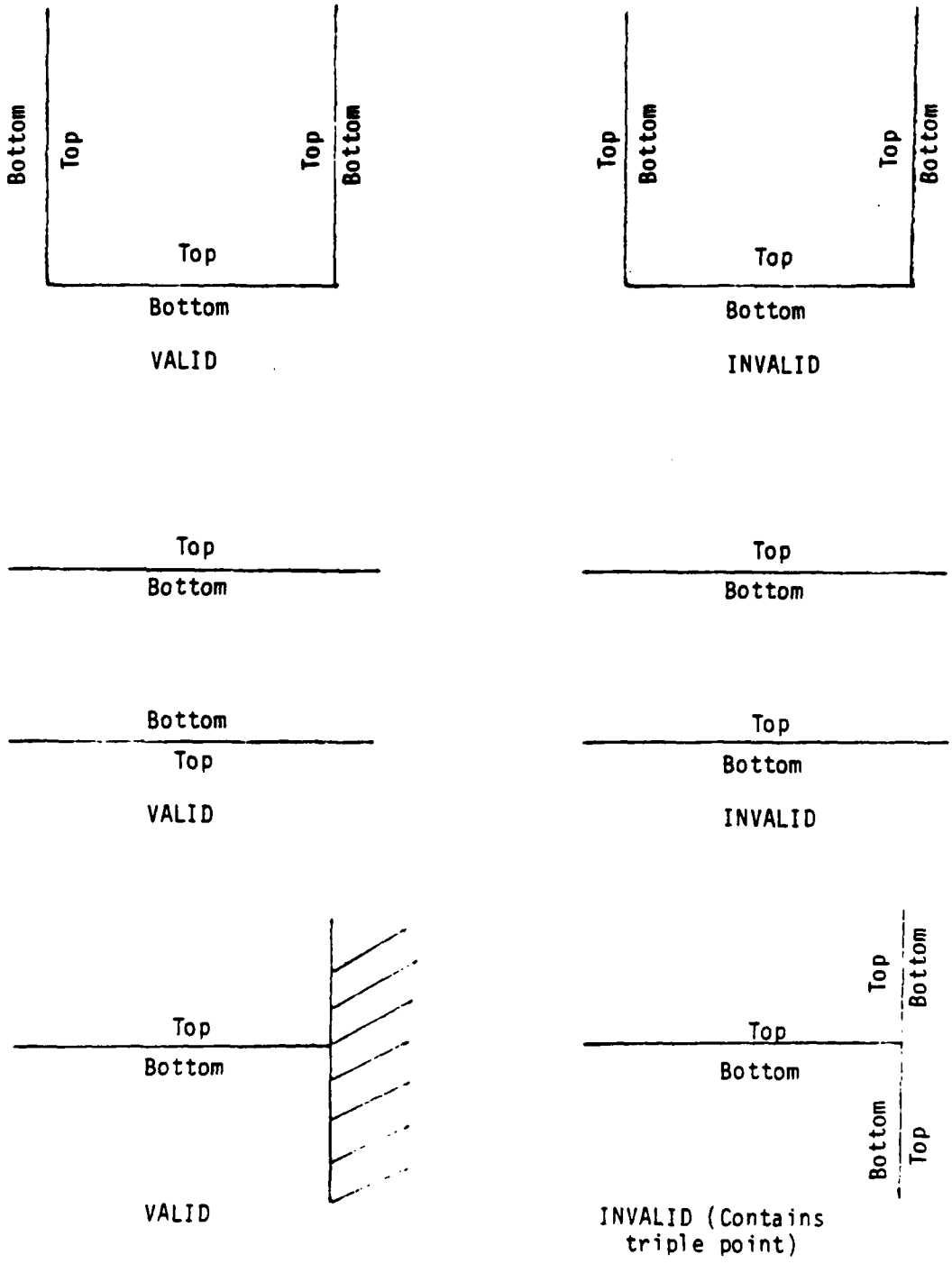


Figure 6/16. Examples of plates intersecting objects.

6.12.10 MATERIAL PROPERTIES

Each material name (e.g., KAPTON, GOLD, FRED (the name is arbitrary)) has associated with it a list of material properties. The name of each material and the values for each material property are supplied by the user in the object definition file. (This is explained in Section 6.12.11.) The nineteen material properties are summarized in Table 6/5. Here we examine each one in more detail.

DIELECTRIC CONSTANT (PROPERTY 1)

Property 1 contains the relative dielectric constant for an insulating material ϵ_r

$$\epsilon_r = \frac{\epsilon}{\epsilon_0}$$

where ϵ is the absolute dielectric constant and ϵ_0 is the dielectric constant of free space. ϵ_r is dimensionless.

THICKNESS (PROPERTY 2)

Property 2 gives the thickness d of a dielectric film covering an underlying conductor in meters. d is arbitrary and may be chosen to be more or less than a mesh unit. However, note that POLAR uses thin-film approximations in many of its calculations involving surfaces (Section 4.52).

BULK CONDUCTIVITY (PROPERTY 3)

Property 3 gives the bulk conductivity σ_0 of the surface material in $\text{ohms}^{-1}\ \text{m}^{-1}$. σ_0 is assumed to be the value appropriate for a sample not exposed to any radiation and not subject to any internal electric fields. Field enhancement and radiation enhancement of σ_0 are not currently modeled by POLAR. A value of -1 indicates that the material is a metallic conductor.

TABLE 6/5. MATERIAL PROPERTIES
(see Section 6.12.10 for notes)

Property No.	User Input Units	Description
1	None	Relative dielectric constant
2	m	Dielectric material thickness
3	$ohms^{-1}\ m^{-1}$	Bulk conductivity (= -1 for a metallic conductor)
4	None	Atomic number
5	None	Maximum secondary electron yield for electron impact
6	keV	Primary electron energy that produces maximum secondary yield
7	angstroms	Range parameters (4.3)
8	None	$R = P_7 E^{P_8} + P_9 E^{P_{10}}$
9	angstroms	
10	None	
11	None	Secondary electron yield due to impact of 1 keV protons
12	keV	Incident proton energy that produces maximum secondary electron yield
13	$A\ m^{-2}$	Photoelectron yield for normally incident sunlight
14	$ohms\ square^{-1}$	Surface resistivity (= -1 for non-conducting surface)
15	Volts	Maximum (absolute) potential attainable before a discharge must occur
16	Volts	Maximum potential difference between surface and underlying conductor before a discharge must occur
17	$ohms^{-1}\ m^{-1}$ $(m^2\ s^{-3})^{-1}$	Radiation-induced conductivity coefficient (k)
18	None	Radiation-induced conductivity power (Δ)
19	$kg\ m^{-3}$	Material density

ATOMIC NUMBER (PROPERTY 4)

Property 4 is the atomic number for pure elements or the mean atomic number for chemical compounds; e.g., polyethylene $(CH_2)_n$ has a mean atomic number of $(6 + 1 + 1)/3 = 2.7$.

SECONDARY YIELD (PROPERTIES 5 AND 6)

Properties 5 and 6 are the coordinates of the maximum in the secondary electron yield curve of the material. The secondary yield curve is a plot of secondary yield δ

$$\delta = \frac{\text{current of secondary electrons emitted}}{\text{incident primary electron current}}$$

for normally incident electrons, against the incident energy of the primary electron E. This is further discussed and illustrated in Section 4.52. Property 5 contains δ_{max}, and property 6 contains E_{max} in keV.

ELECTRON RANGE (PROPERTIES 7, 8, 9 AND 10)

Part of the secondary electron emission formulation requires an analytical form for the "range" of electrons in the material. The range is the depth to which the electrons can penetrate the material as they are continuously slowed down by losing energy to the material lattice. POLAR uses a biexponential form. If P_7, P_8, P_9, and P_{10} are properties 7-10 respectively, the range R is given by

$$R = P_7 E^{P_8} + P_9 E^{P_{10}}$$

The four parameters are obtained from fits to stopping power data (Section 4.52). The range is determined in $\mathring{A}$ (10^{-10} m). If no

reliable stopping power data or four parameter fits are available, the range may be estimated from Feldman's formula[4] automatically by assigning -1 to property 7. In this mode properties 7-10 are assigned as follows:

P_7 = -1
P_8 = null
P_9 = material density (g cm^{-3})
P_{10} = mean atomic weight (AMU)

The mean atomic weight is calculated in the same way as the mean atomic number (property 4) using atomic masses rather than numbers.

ION INDUCED SECONDARY EMISSION (PROPERTIES 11 AND 12)

Secondary emission of electrons due to ion impact is also treated using a two parameter theory (4.52). Parameter 11 contains the yield for 1 keV normally incident protons and parameter 12 the proton energy that produces the maximum electron yield. The secondary emission properties due to impact of ions other than protons are assumed to be identical to the proton values.

PHOTOEMISSION (PROPERTY 13)

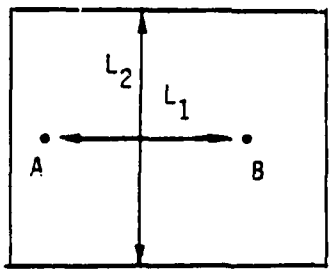
Property 13 contains the yield of photoelectrons from the surface material exposed to the solar spectrum. The intensity is that measured on earth 93,000,000 miles from the sun. (Earth orbit altitudes are negligible by comparison and the intensity of the sun close to earth may be considered constant.)

SURFACE RESISTIVITY (PROPERTY 14)

Property 14 gives the intrinsic surface resistivity in the "ohms per square". This rather odd unit is used to distinguish the resistivity coefficient (property 14) from the actual surface resistance (in ohms) calculated by POLAR. Consider two points in a plane A and B, a distance L_1 apart. If L_2 is the "width" of the plane

$$\text{surface resistance} = \text{surface resistivity} \times \frac{L_1}{L_2}$$

$$\text{i.e.} \qquad \text{ohms} = (\text{ohms per square}) \times \begin{matrix}\text{dimensionless} \\ \text{geometrical} \\ \text{factor}\end{matrix}$$



POLAR uses the surface resistivity per square, times a geometrical factor it calculates to determine the surface resistance between two adjacent materials.

The intrinsic surface conductivity is due to the migration of electrons along the surface layer aided by adsorbed impurities and defects.

Surface conductivity may be omitted from the current calculations completely by choosing property 14 to be negative.

DISCHARGE ANALYSIS (PROPERTIES 17, 18, 19, 20)

Currently under development.

PROPERTIES 17, 18, 19, 20 (RADIATION INDUCED CONDUCTIVITY)

Currently under development.

6.12.11 DEFINING MATERIALS

New materials are defined, and their properties assigned inside the object definition file (6.10.11). The object definition file is read by POLAR module VEHICL. VEHICL interprets any word that it does not recognize as a building block keyword (or their parameter cards (6.10.25)) as the definition of a new material name. New material names may <u>not</u> appear inside building block definitions (i.e., between a building block keyword and an 'ENDOBJ' statement).

Following the material name, VEHICL expects to find three additional cards specifying 20 constants as the material properties to be associated with the name. The 20 constants correspond to properties 1-20 and are read sequentially; i.e., the first constant read is interpreted as property 1, the second, property 2, and so on. They are arranged sequentially, eight per card, so that cards 1 and 2 each have eight numbers and card 3, four numbers. Formally each number is written in a field of up to ten characters, but POLAR will read the cards in free format. No units need be specified. POLAR will assume the units given in Table 6/5 and no others. For properties not requiring any input such as property 20, or properties 17-19 for conductors, some constant must be entered but its value is arbitrary. (POLAR will not actually use the values entered but expects to read something.)

Once the three material property cards have been read VEHICL is ready to accept any other keywords or more material names. POLAR will recognize up to fifteen different materials.

Materials must be defined before they are referred to in any building block definition. For example, if I assign the surface of a sphere to be 'FSTUFF' with the card

MATERIAL FSTUFF

if 'FSTUFF' and its material properties have not been declared earlier in the object definition file, an error will occur and execution will stop. For this reason all the materials to be used are usually declared at the very beginning of the object definition file. This is shown in Figure 6/17.

```
COMMENT  DEFINITION OF SATELLITE "BIG EARS"

Material Name 1
  3 material property cards

Material Name 2
  3 material property cards
    .
    .
    .

COMMENT  DEFINE MAIN BODY

CONDUCTOR 1

QSPHERE
  parameter cards
ENDOBJ

RECTAN
  parameter cards
ENDOBJ

    .
    .    more building blocks
    .

COMMENT  DEFINE SOLAR PANEL (SEPARATE CONDUCTOR)

CONDUCTOR 2

PLATE
  parameter cards
ENDOBJ

    .
    .   more building blocks
    .

COMMENT

CONDUCTOR 3
    .
    .   more conductor segments
    .

ENDSAT
```

Figure 6/17. General form of the object definition file.

6.12.12 DEFAULT MATERIALS

There is one case where the user can forget to define his or her materials and get away with it. When VEHICL encounters a material that hasn't been defined already, before an error occurs, it checks the following list of default materials:

ALUMIN
AQUADG
CPAINT
GOLD
INDOX
MAGNES
SCREEN
KAPTON
NPAINT
SIO2
SOLAR
TEFLON
SILVER

If the material is included in this list, it becomes one of the up to fifteen defined materials and its properties, stored internally, are automatically entered as VEHICL input by the code. The properties of these materials are shown in Table 6/6. Any further reference to the material will assign the same set of properties to the surfaces concerned. If the material is not found in this list, an error will occur. These material properties are currently a carryover from NASCAP. New default materials will be included in future revisions.

If two sets of material properties are defined with the same name, or names with the same first four letters, two of the fifteen possible materials are used up but only the first set of material properties are used. For example, if GOLD is referenced before it is defined in the runstream, the default material properties of gold will be associated with all gold surfaces in the object definition file.

TABLE 6/6
MATERIAL PROPERTIES

MATERIAL 1: ALUM

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	1.00+000 (NONE)	1.00+000 (NONE)
2	THICKNESS	1.00-003 METERS	1.00-002 MESH
3	CONDUCTIVITY	-1.00+000 MHO/M	-1.00+000 MHO/M
4	ATOMIC NUMBER	1.30+001 (NONE)	1.30+001 (NONE)
5	DELTA MAX >COEFF	9.70-001 (NONE)	9.18+000 (NONE)
6	E-MAX >DEPTH**-1	3.00-001 KEV	3.00-002 ANG-01
7	RANGE	1.54+002 ANG.	1.23+002 ANG.
8	EXPONENT > RANGE	9.00-001 (NONE)	3.87+002 ANG.
9	RANGE > EXPONENT	2.20+002 ANG.	6.00-001 (NONE)
10	EXPONENT	1.76+000 (NONE)	1.76+000 (NONE)
11	YIELD FOR 1KEV PROTONS	2.44-001 (NONE)	2.44-001 (NONE)
12	MAX DE/DX FOR PROTONS	2.30+002 KEV	2.30+002 KEV
13	PHOTOCURRENT	4.00-005 A/M**2	4.00-005 A/M**2
14	SURFACE RESISTIVITY	-1.00+000 OHMS	-8.85-013 V-S/C
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMSJ	1.00-013 MHOMSJ
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	-1.00+000

MATERIAL 2: AQUA

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	1.00+000 (NONE)	1.00+000 (NONE)
2	THICKNESS	1.00-003 METERS	1.00-002 MESH
3	CONDUCTIVITY	-1.00+000 MHO/M	-1.00+000 MHO/M
4	ATOMIC NUMBER	8.00+000 (NONE)	8.00+000 (NONE)
5	DELTA MAX >COEFF	1.00+000 (NONE)	7.36+000 (NONE)
6	E-MAX >DEPTH**-1	3.00-001 KEV	2.21-002 ANG-01
7	RANGE	-1.00+000 ANG.	5.00+002 ANG.
8	EXPONENT > RANGE	.00 (NONE)	.00 ANG.
9	RANGE > EXPONENT	2.00+000 ANG.	1.55+000 (NONE)
10	EXPONENT	1.20+001 (NONE)	1.00+000 (NONE)
11	YIELD FOR 1KEV PROTONS	4.55-001 (NONE)	4.55-001 (NONE)
12	MAX DE/DX FOR PROTONS	1.40+002 KEV	1.40+002 KEV
13	PHOTOCURRENT	2.10-005 A/M**2	2.10-005 A/M**2
14	SURFACE RESISTIVITY	-1.00+000 OHMS	-8.85-013 V-S/C
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMSJ	1.00-013 MHOMSJ
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	-1.00+000

TABLE 6/6

MATERIAL PROPERTIES (Continued)

MATERIAL 3: CPAI

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	3.50+000 (NONE)	3.50+000 (NONE)
2	THICKNESS	1.00-003 METERS	1.00-002 MESH
3	CONDUCTIVITY	-1.00+000 MHO/M	-1.00+000 MHO/M
4	ATOMIC NUMBER	5.00+000 (NONE)	5.00+000 (NONE)
5	DELTA MAX >COEFF	2.10+000 (NONE)	4.06+001 (NONE)
6	E-MAX >DEPTH**-1	1.30-001 KEV	8.74-002 ANG-01
7	RANGE	7.15+001 ANG.	4.29+001 ANG.
8	EXPONENT > RANGE	6.00-001 (NONE)	5.62+002 ANG.
9	RANGE > EXPONENT	3.12+002 ANG.	6.00-001 (NONE)
10	EXPONENT	1.77+000 (NONE)	1.77+000 (NONE)
11	YIELD FOR 1KEV PROTONS	4.55-001 (NONE)	4.55-001 (NONE)
12	MAX DE/DX FOR PROTONS	1.40+002 KEV	1.47+002 KEV
13	PHOTOCURRENT	2.00-005 A/M**2	2.00-005 A/M**2
14	SURFACE RESISTIVITY	-1.00+000 OHMS	-8.85-013 V-S/Q
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMSS	1.00-013 MHOMSS
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	-1.00+000

MATERIAL 4: GOLD

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	1.00+000 (NONE)	1.00+000 (NONE)
2	THICKNESS	1.00-003 METERS	1.00-002 MESH
3	CONDUCTIVITY	-1.00+000 MHO/M	-1.00+000 MHO/M
4	ATOMIC NUMBER	7.90+001 (NONE)	7.90+001 (NONE)
5	DELTA MAX >COEFF	8.80-001 (NONE)	2.43+000 (NONE)
6	E-MAX >DEPTH**-1	5.00-001 KEV	2.02-002 ANG-01
7	RANGE	6.89+001 ANG.	8.17+001 ANG.
8	EXPONENT > RANGE	9.00-001 (NONE)	9.25+001 ANG.
9	RANGE > EXPONENT	5.35+001 ANG.	9.00-001 (NONE)
10	EXPONENT	1.73+000 (NONE)	1.73+000 (NONE)
11	YIELD FOR 1KEV PROTONS	4.13-001 (NONE)	4.13-001 (NONE)
12	MAX DE/DX FOR PROTONS	1.35+002 KEV	1.35+002 KEV
13	PHOTOCURRENT	2.90-005 A/M**2	2.90-005 A/M**2
14	SURFACE RESISTIVITY	-1.00+000 OHMS	-8.85-013 V-S/Q
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMSS	1.00-013 MHOMSS
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	-1.00+000

TABLE 6/6

MATERIAL PROPERTIES (Continued)

MATERIAL 5: INOO

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	1.00+000 (NONE)	1.00+000 (NONE)
2	THICKNESS	1.00-003 METERS	1.00-002 MESH
3	CONDUCTIVITY	-1.00+000 MHO/M	-1.00+000 MHO/M
4	ATOMIC NUMBER	2.44+001 (NONE)	2.44+001 (NONE)
5	DELTA MAX >COEFF	1.40+000 (NONE)	3.32+000 (NONE)
6	E-MAX >DEPTH**-1	8.00-001 KEV	1.49-002 ANG-01
7	RANGE	-1.00+000 ANG.	1.67+002 ANG.
8	EXPONENT > RANGE	.00 (NONE)	.00 ANG.
9	RANGE > EXPONENT	7.18+000 ANG.	2.01+000 (NONE)
10	EXPONENT	5.55+001 (NONE)	1.00+000 (NONE)
11	YIELD FOR 1KEV PROTONS	4.90-001 (NONE)	4.90-001 (NONE)
12	MAX DE/DX FOR PROTONS	1.23+002 KEV	1.23+002 KEV
13	PHOTOCURRENT	3.20-005 A/M**2	3.20-005 A/M**2
14	SURFACE RESISTIVITY	-1.00+000 OHMS	-8.85-012 V-S/C
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMS3	1.00-013 MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	-1.00+000

MATERIAL 6: MAGN

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	1.00+000 (NONE)	1.00+000 (NONE)
2	THICKNESS	1.00-003 METERS	1.00-002 MESH
3	CONDUCTIVITY	-1.00+000 MHO/M	-1.00+000 MHO/M
4	ATOMIC NUMBER	1.20+001 (NONE)	1.20+001 (NONE)
5	DELTA MAX >COEFF	9.20-001 (NONE)	7.02+000 (NONE)
6	E-MAX >DEPTH**-1	2.60-001 KEV	2.79-002 ANG-01
7	RANGE	-1.00+000 ANG.	6.96+002 ANG.
8	EXPONENT > RANGE	.00 (NONE)	.00 ANG.
9	RANGE > EXPONENT	1.74+000 ANG.	1.75+000 (NONE)
10	EXPONENT	2.43+001 (NONE)	1.00+000 (NONE)
11	YIELD FOR 1KEV PROTONS	2.44-001 (NONE)	2.44-001 (NONE)
12	MAX DE/DX FOR PROTONS	2.30+002 KEV	2.30+002 KEV
13	PHOTOCURRENT	4.00-005 A/M**2	4.00-005 A/M**2
14	SURFACE RESISTIVITY	-1.00+000 OHMS	-8.85-012 V-S/C
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMS3	1.00-013 MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	-1.00+000

TABLE 6/6
MATERIAL PROPERTIES (Continued)

MATERIAL 7: SCRE

	PROPERTY	INPUT VALUE		CODE VALUE	
1	DIELECTRIC CONSTANT	1.00+000	(NONE)	1.00+000	(NONE)
2	THICKNESS	1.00-003	METERS	1.00-002	MESH
3	CONDUCTIVITY	-1.00+000	MHO/M	-1.00+000	MHO/M
4	ATOMIC NUMBER	1.00+000	(NONE)	1.00+000	(NONE)
5	DELTA MAX >COEFF	.00	(NONE)	.00	(NONE)
6	E-MAX >DEPTH**-1	1.00+000	KEV	1.38-001	ANG-01
7	RANGE	1.00+001	ANG.	1.50+001	ANG.
8	EXPONENT > RANGE	1.50+000	(NONE)	.00	ANG.
9	RANGE > EXPONENT	.00	ANG.	1.50+000	(NONE)
10	EXPONENT	1.00+000	(NONE)	1.00+000	(NONE)
11	YIELD FOR 1KEV PROTONS	.00	(NONE)	.00	(NONE)
12	MAX DE/DX FOR PROTONS	1.00+000	KEV	1.00+000	KEV
13	PHOTOCURRENT	.00	A/M**2	.00	A/M**2
14	SURFACE RESISTIVITY	-1.00+000	OHMS	-9.85-013	V-S/C
15	SPACE DISCHARGE POT'L	1.00+004	VOLTS	1.00+004	VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003	VOLTS	2.00+003	VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013	MHOMS3	1.00-013	MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000	(NONE)	1.00+000	(NONE)
19	DENSITY	1.00+003	KG/M**3	1.00+003	KG/M**3
20		2.00+001		-1.00+000	

MATERIAL 8: KAPT

	PROPERTY	INPUT VALUE		CODE VALUE	
1	DIELECTRIC CONSTANT	3.50+000	(NONE)	3.50+000	(NONE)
2	THICKNESS	1.27-004	METERS	1.27-003	MESH
3	CONDUCTIVITY	1.00-016	MHO/M	1.00-016	MHO/M
4	ATOMIC NUMBER	5.00+000	(NONE)	5.00+000	(NONE)
5	DELTA MAX >COEFF	2.10+000	(NONE)	4.36+001	(NONE)
6	E-MAX >DEPTH**-1	1.60-001	KEV	8.74-002	ANG-01
7	RANGE	7.15+001	ANG.	4.29+001	ANG.
8	EXPONENT > RANGE	6.00-001	(NONE)	5.62+002	ANG.
9	RANGE > EXPONENT	3.12+002	ANG.	6.00-001	(NONE)
10	EXPONENT	1.77+000	(NONE)	1.77+000	(NONE)
11	YIELD FOR 1KEV PROTONS	4.55-001	(NONE)	4.55-001	(NONE)
12	MAX DE/DX FOR PROTONS	1.40+002	KEV	1.40+002	KEV
13	PHOTOCURRENT	2.00-005	A/M**2	2.00-005	A/M**2
14	SURFACE RESISTIVITY	1.00+016	OHMS	8.85+003	V-S/C
15	SPACE DISCHARGE POT'L	1.00+004	VOLTS	1.00+004	VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003	VOLTS	2.00+003	VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013	MHOMS3	1.00-013	MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000	(NONE)	1.00+000	(NONE)
19	DENSITY	1.00+003	KG/M**3	1.00+003	KG/M**3
20		2.00+001		1.00-016	

TABLE 6/6

MATERIAL PROPERTIES (Continued)

MATERIAL 9: NPAI

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	3.50+000 (NONE)	3.50+000 (NONE)
2	THICKNESS	5.00-005 METERS	5.00-004 MESH
3	CONDUCTIVITY	5.90-014 MHO/M	5.90-014 MHO/M
4	ATOMIC NUMBER	5.00+000 (NONE)	5.00+000 (NONE)
5	DELTA MAX >COEFF	2.10+000 (NONE)	3.05+001 (NONE)
6	E-MAX >DEPTH**-1	1.60-001 KEV	3.41-002 ANG-01
7	RANGE	-1.00+000 ANG.	1.05+003 ANG.
8	EXPONENT > RANGE	.00 (NONE)	.00 ANG.
9	RANGE > EXPONENT	1.05+000 ANG.	1.51+000 (NONE)
10	EXPONENT	9.80+000 (NONE)	1.00+000 (NONE)
11	YIELD FOR 1KEV PROTONS	4.55-001 (NONE)	4.55-001 (NONE)
12	MAX DE/DX FOR PROTONS	1.40+002 KEV	1.40+002 KEV
13	PHOTOCURRENT	2.00-005 A/M**2	2.00-005 A/M**2
14	SURFACE RESISTIVITY	1.00+013 OHMS	9.85+000 V-S/C
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMS3	1.00-013 MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	5.90-014

MATERIAL 10: SIO2

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	4.00+000 (NONE)	4.00+000 (NONE)
2	THICKNESS	1.27-004 METERS	1.27-003 MESH
3	CONDUCTIVITY	1.00-014 MHO/M	1.00-014 MHO/M
4	ATOMIC NUMBER	1.00+001 (NONE)	1.00+001 (NONE)
5	DELTA MAX >COEFF	2.40+000 (NONE)	1.46+001 (NONE)
6	E-MAX >DEPTH**-1	4.30-001 KEV	2.21-002 ANG-01
7	RANGE	1.16+002 ANG.	9.42+001 ANG.
8	EXPONENT > RANGE	8.10-001 (NONE)	3.41+002 ANG.
9	RANGE > EXPONENT	1.83+002 ANG.	8.10-001 (NONE)
10	EXPONENT	1.86+000 (NONE)	1.86+000 (NONE)
11	YIELD FOR 1KEV PROTONS	4.55-001 (NONE)	4.55-001 (NONE)
12	MAX DE/DX FOR PROTONS	1.40+002 KEV	1.40+002 KEV
13	PHOTOCURRENT	2.00-005 A/M**2	2.00-005 A/M**2
14	SURFACE RESISTIVITY	1.00+019 OHMS	8.85+006 V-S/C
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMS3	1.00-013 MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	1.00-014

TABLE 6/6

MATERIAL PROPERTIES (Continued)

MATERIAL 11: SOLA

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	3.80+000 (NONE)	3.80+000 (NONE)
2	THICKNESS	1.79-004 METERS	1.79-003 MESH
3	CONDUCTIVITY	1.00-017 MHO/M	1.00-017 MHO/M
4	ATOMIC NUMBER	1.00+001 (NONE)	1.00+001 (NONE)
5	DELTA MAX >COEFF	2.05+000 (NONE)	1.31+001 (NONE)
6	E-MAX >DEPTH**-1	4.10-001 KEV	3.17-002 ANG-01
7	RANGE	7.75+001 ANG.	3.49+001 ANG.
8	EXPONENT > RANGE	4.50-001 (NONE)	2.70+002 ANG.
9	RANGE > EXPONENT	1.56+002 ANG.	4.50-001 (NONE)
10	EXPONENT	1.73+000 (NONE)	1.73+000 (NONE)
11	YIELD FOR 1KEV PROTONS	2.44-001 (NONE)	2.44-001 (NONE)
12	MAX DE/DX FOR PROTONS	2.30+002 KEV	2.30+002 KEV
13	PHOTOCURRENT	2.00-005 A/M**2	2.00-005 A/M**2
14	SURFACE RESISTIVITY	1.00+019 OHMS	8.85+006 V-S/Q
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMS3	1.00-013 MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	1.00-016

MATERIAL 12: TEFL

	PROPERTY	INPUT VALUE	CODE VALUE
1	DIELECTRIC CONSTANT	2.00+000 (NONE)	2.00+000 (NONE)
2	THICKNESS	1.27-004 METERS	1.27-003 MESH
3	CONDUCTIVITY	1.00-016 MHO/M	1.00-016 MHO/M
4	ATOMIC NUMBER	7.00+000 (NONE)	7.00+000 (NONE)
5	DELTA MAX >COEFF	3.00+000 (NONE)	2.27+001 (NONE)
6	E-MAX >DEPTH**-1	3.00-001 KEV	3.83-002 ANG-01
7	RANGE	4.54+001 ANG.	1.81+001 ANG.
8	EXPONENT > RANGE	4.00-001 (NONE)	3.65+002 ANG.
9	RANGE > EXPONENT	2.18+002 ANG.	4.00-001 (NONE)
10	EXPONENT	1.77+000 (NONE)	1.77+000 (NONE)
11	YIELD FOR 1KEV PROTONS	4.55-001 (NONE)	4.55-001 (NONE)
12	MAX DE/DX FOR PROTONS	1.40+002 KEV	1.40+002 KEV
13	PHOTOCURRENT	2.00-005 A/M**2	2.00-005 A/M**2
14	SURFACE RESISTIVITY	1.00+016 OHMS	8.85+003 V-S/Q
15	SPACE DISCHARGE POT'L	1.00+004 VOLTS	1.00+004 VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003 VOLTS	2.00+003 VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013 MHOMS3	1.00-013 MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000 (NONE)	1.00+000 (NONE)
19	DENSITY	1.00+003 KG/M**3	1.00+003 KG/M**3
20		2.00+001	1.00-016

TABLE 6/6
MATERIAL PROPERTIES (Concluded)

MATERIAL 13: SILV

	PROPERTY	INPUT VALUE		CODE VALUE	
1	DIELECTRIC CONSTANT	1.00+000	(NONE)	1.00+000	(NONE)
2	THICKNESS	1.00-003	METERS	1.00-002	MESH
3	CONDUCTIVITY	-1.00+000	MHO/M	-1.00+000	MHO/M
4	ATOMIC NUMBER	4.70+001	(NONE)	4.70+001	(NONE)
5	DELTA MAX >COEFF	1.00+000	(NONE)	3.09+000	(NONE)
6	E-MAX >DEPTH**-1	8.00-001	KEV	1.58-002	ANG-01
7	RANGE	8.45+001	ANG.	6.93+001	ANG.
8	EXPONENT > RANGE	9.20-001	(NONE)	1.36+002	ANG.
9	RANGE > EXPONENT	7.94+001	ANG.	8.20-001	(NONE)
10	EXPONENT	1.74+000	(NONE)	1.74+000	(NONE)
11	YIELD FOR 1KEV PROTONS	4.90-001	(NONE)	4.90-001	(NONE)
12	MAX DE/DX FOR PROTONS	1.23+002	KEV	1.23+002	KEV
13	PHOTOCURRENT	2.90-005	A/M**2	2.90-005	A/M**2
14	SURFACE RESISTIVITY	-1.00+000	OHMS	-6.65-013	V-S/C
15	SPACE DISCHARGE POT'L	1.00+004	VOLTS	1.00+004	VOLTS
16	INTERNAL DISCHARGEPOT'L	2.00+003	VOLTS	2.00+003	VOLTS
17	RADN INDUCEDCOND'YCOEFFT	1.00-013	MHOMS3	1.00-013	MHOMS3
18	RADN INDUCEDCOND'YPOWER	1.00+000	(NONE)	1.00+000	(NONE)
19	DENSITY	1.00+003	KG/M**3	1.00+003	KG/M**3
20		2.00+001		-1.00+000	

If a material called 'GOLD' or 'GOLDPD' or 'GOLDXXXX' is defined later with different properties the number of materials POLAR thinks it has will be increased by one, but the new properties will be effectively ignored. Multiple definition of materials should be avoided. Note, however, that if any of the default materials are explicitly defined before they are referred to in building block definitions then POLAR will make no attempt to find them in the list of default materials and the materials will not be multiple defined.

6.13 THE OBJECT DEFINITION FILE - ANOTHER EXAMPLE

We are now ready to bring together Sections 6.10-6.12 and examine the structure of the object definition file. The general form is shown in Figure 6/17. The materials are defined first, followed by the building blocks associated with each separate conductor. The use of COMMENT cards allow the logic of the definition of a complex object to be followed more easily. Finally the whole file is terminated with an 'ENDSAT' statement. An actual example is shown in Figure 6/18. It consists of a central RECTANgular body connected to two QSPHERES at the ends.

A 3D-VIEW (6.20) of the object produced by VEHICL is shown in Figure 6/19.

```
 1:COMMENT WORKED EXAMPLE (SECTION 6.13)
 2:COMMENT    MATERIAL DEFINITIONS
 3:COMMENT    FOR PRESENT PURPOSES, ALL PROPERTIES ARE 'KAPTON'
 4:COMMENT    EXCEPT THICKER
 5:TKAP
 6:       3.5,.01,1.E-16,5.,2.1,.15,71.48,.60,
 7:       312.1,1.77,.455,140.,.00002,1.E+16,1.E+4,2.E+3,
 8:       1.E-13,1.,1.E+3,20.
 9:COMMENT USE DEFAULT KAPTON AND GOLD VALUES
10:COMMENT DEFINE THE MAIN BODY AND TOP SPHERE TO BE ON
11:CONDUCTOR 1
12:COMMENT MAIN CUBOID BODY
13:RECTAN
14:CORNER -1 -1 -2
15:DELTAS 3 3 4
16:SURFACE +X GOLD
17:SURFACE -X KAPTON
18:SURFACE -Y KAPTON
19:SURFACE +Y KAPTON
20:SURFACE -Z KAPTON
21:SURFACE +Z TKAP
22:ENDOBJ
23:COMMENT TKAP SPHERE ON TOP
24:QSPHERE
25:CENTER 0 0 3
26:DIAMETER 3
27:SIDE 1
28:MATERIAL TKAP
29:ENDOBJ
30:COMMENT PUT THE BOTTOM SPHERE ON CONDUCTOR 2
31:CONDUCTOR 2
32:QSPHERE
33:CENTER 0 0 -4
34:DIAMETER 3
35:SIDE 1
36:MATERIAL GOLD
37:ENDOBJ
38:ENDSAT
EOF:38
0:>
```

Figure 6/18. Object definition file.

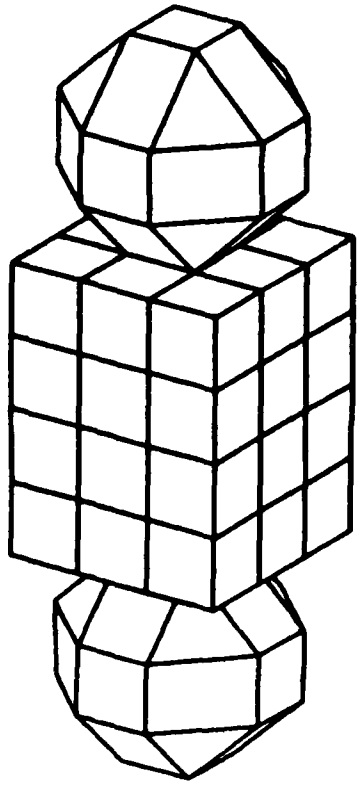


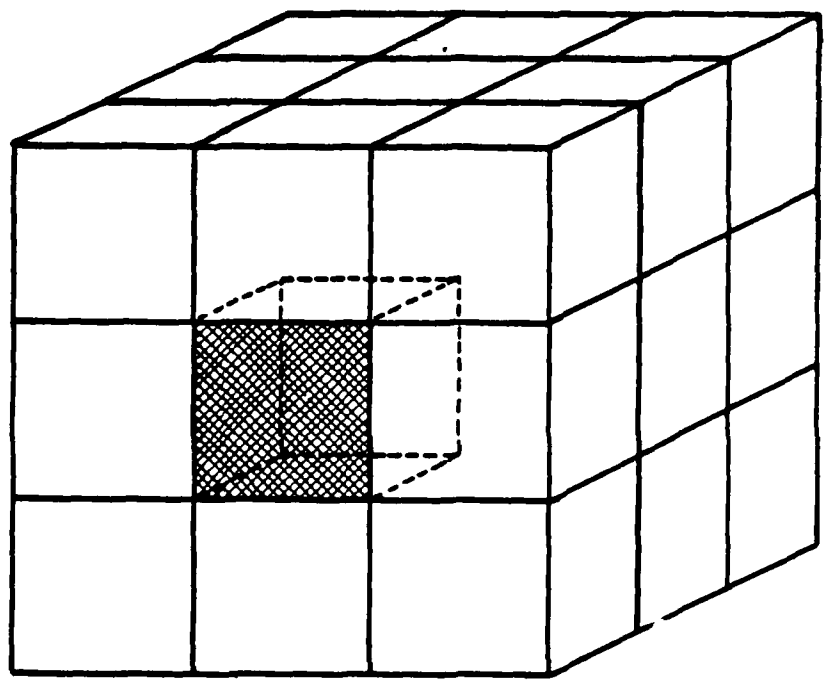
Figure 6/19. 3-D view of object produced by HIDCEL (hidden lines).

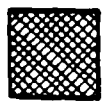
6.14 OBJECTS WITHIN OBJECTS: VARIEGATED SURFACES

POLAR makes it easy to define surfaces that are made up of more than one material. For example, we may want to define one face of a cube to be mainly KAPTON but with a patch of say GOLD in the center (Figure 6/20). We begin by defining the cube with a KAPTON face. The center surface cell is then replaced with GOLD by defining a second smaller cube inside the first cube. The second cube is defined so that one of its faces is coincident with the KAPTON face. The surface common to both cubes is then associated with the material on the face of the second cube, which in this case is GOLD. This is shown in Figure 6/20.

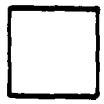
The object definition file associated with this object has the form:

```
COMMENT                 VARIEGATED CUBE
RECTAN
CORNER                  -2  -2  -2
DELTAS                   3   3   3
SURFACE                 +x  KAPTON
SURFACE                 -x  KAPTON
SURFACE                 +y  KAPTON
SURFACE                 -y  KAPTON
SURFACE                 +z  KAPTON
SURFACE                 -z  KAPTON
ENDOBJ
RECTAN                  -1  -1  -1
CORNER
DELTA                    1   1   1
SURFACE                 -z  GOLD
ENDOBJ
ENDSAT
```





GOLD



KAPTON

Figure 6/20. A variegated surface definition.

The same principle can be applied to any of the building blocks. Exposed surface cells common to two or more building blocks are assigned to the material of the most recently defined block.

Two special building blocks are supplied specifically to create variegated surfaces. PATCHR and PATCHW define a RECTAN (cuboid) and a WEDGE respectively, that may be used to "patch" other objects without adding to POLAR's list of filled space. The use of actual RECTAN and WEDGE blocks inside others is also perfectly legitimate, but adds to the internally used surface list. The use of PATCHR and PATCHW reduces the likelihood of a problem occurring due to the list overflowing.

The object shown in Figure 6/20 could also be defined using PATCHR:

```
COMMENT     VARIEGATED CUBE (PATCHR)
RECTAN
CORNER      -2  -2  -2
DELTAS       3   3   3
SURFACE     +X  KAPTON
SURFACE     -X  KAPTON
SURFACE     +Y  KAPTON
SURFACE     -Y  KAPTON
SURFACE     +Z  KAPTON
SURFACE     -Z  KAPTON
ENDOBJ
PATCHR
CORNER      -1  -1  -1
DELTAS       1   1   1
SURFACE     -Z  GOLD
ENDOBJ
ENDSAT
```

6.20 VEHICL

The VEHICL module is used to interpret the object definition file (Section 6.10) in order to create the various tables and lists necessary for the other modules (Sections 5.23-5.25). It also can produce two separate kinds of object plots. By using appropriate keywords, either material or perspective plots are produced after the object file passes through the initial definition processing.

When using keyword inputs, each line (or card) is expected to contain one keyword followed by its list of parameters. After the keyword and parameters have been completely defined, the rest of the line is ignored. Several characters are ignored by the input routines, namely extra blanks between words (though some type of delimeter is necessary), equal signs (=), and commas (,). All input is read using free formatting with lower case characters being converted to upper case, and real numbers may be entered as integers.

VEHICL will need the following permanent files: 2. (for graphics output), 11. (MSIO), 19. (MSIO), and 20. (the object definition file, used as input). The following temporary scratch files will also be required: 3., 14., 17., 18., 21., and 27.. All of these files should be assigned with large storage limits.

6.21 VEHICL KEYWORDS

The specialized VEHICL keywords fall into three general categories; object definition, graphical output, and diagnostic output control. (See Table 6.21/1 for a brief summary of the keywords.) The diagnostic keywords are described in detail in Section 6.22. Additionally, all of the general POLAR keywords (Section 6.70), except SELECT, are recognized by the input routines. SELECT cannot be used by VEHICL because the grid information needed by the subroutine, MRBUF (5.30), is not necessarily well defined.

If an unrecognized keyword or an invalid use of a keyword is discovered, VEHICL issues a warning or an error message then terminates batch runs or reads the next input in the case of interactive runs. By default, VEHICL believes it is being run interactively. The keyword BATCH (see Section 6.70) will place VEHICL in its batch input mode.

The recognized keywords which control the object and grid definition are:

NXYZ

NXYZ defines the size of the object grid. This keyword absolutely must be defined for each execution of VEHICL since there are no default values for grid dimensions. The run will unconditionally abort in the absence of this keyword. An example of the use of the keyword is

NXYZ 6 7 8

This defines an object grid which has six nodes in the X-direction, seven nodes along the Y axis, and eight on the Z edge of the grid. Note that these numbers are in nodes, not elements. So a 1 x 1 x 1 cube would require a grid of at least 4 x 4 x 4 since object itself is 2 x 2 x 2 nodes and none of the object's vertices are allowed to touch the grid boundary. In general, extra space around the object is a good idea, especially if the object is centered asymmetrically in the grid and the object may be reoriented within the object grid later using ORIENT (6.30). These problems can also be avoided if an odd number of nodes along each axis are used. For more information concerning the declaration of object grid size, please see Section 6.10.

DXMESH

DXMESH defines grid size in meters. For example,

DXMESH = 2.0

would define a grid spacing of two meters, while

DXMESH .01

sets the spacing to one centimeter. The default value is one meter. This size may be easily changed later during NTERAK.

PREFIX

PREFIX is used to declare the object file name for the UNIVAC version of POLAR. The CYBER version expects the object definition to be located in file 2Q, so the PREFIX keyword is not needed.

In the UNIVAC version, VEHICL must know the file's name in order to be able to find the input. If this keyword is omitted in the UNIVAC version, VEHICL will terminate with an error message. As an example,

PREFIX MICRO

causes VEHICL to read from the file MICROOBJ. Note the suffix OBJ must be used in all object file names. There is no default value for this keyword.

GRAPHICAL OUTPUT CONTROL KEYWORDS

MATPLOTS

The keyword MATPLOTS controls the plotting of material plots. Material plots consist of six views of the object from each direction of all three axes. The surfaces of the satellite are filled in with different patterns depending on their material type. It should be noted only VEHICL can produce material plots, SHONTL is not able to duplicate them, although it can draw the same views without surface material patterns.

To control this feature, use

MATPLOTS YES

to turn it on and

MATPLOTS NO

to turn it off. By default, no material plots will be created.

The graphical output created by VEHICL is written on file 2, and needs to be interpreted by the post-processor graphics package (see Section 6.6). On the UNIVAC, this can be done automatically by using the PLOTDEST keyword described below.

MAKEPLOT

MAKEPLOT controls the creation of perspective plots. Perspective plots are views of the object as seen from infinity along a user defined vector. Both hidden line and transparent object drawings are produced. These views only can be drawn by VEHICL; SHONTL is not able to generate them.

The keyword MAKEPLOT tells VEHICL how many views to expect. The actual viewing directions are defined using the PLOTDIR keyword described below. To set the number of views, use

MAKEPLOT N

where N is an integer from 0 to 8. A default set of views (along each direction of all three axes) can be requested using

MAKEPLOT DEFAULT

or just

MAKEPLOT

In both cases, PLOTDIR does not need to be used to define the views.

If no perspective plots are desired,

MAKEPLOT 0

will disable this feature. VEHICL by default will not make perspective plots.

The graphical output created by this command will be saved on file 2 and can be interpreted by the graphics post-processor described in Section 6.6. On the UNIVAC this may be done automatically by using the PLOTDEST command described below.

PLOTDIR

PLOTDIR is used to define viewing vectors for perspective plots. Each use of PLOTDIR describes one of the views from infinity requested by the MAKEPLOT keyword described above. To actually produce a plot, MAKEPLOT must be used. An example of the use of PLOTDIR is

PLOTDIR -2. +1.5 -1.

would be a view along $-2.0\hat{i} + 1.5\hat{j} - 1.0\hat{k}$ from infinity. Default directions can be defined with the MAKEPLOT keyword.

PLOTDEST

The keyword PLOTDEST is used to draw the plots created using the MATPLOTS or MAKEPLOT keywords described above immediately after VEHICL completes its execution. The keyword is only functional in the UNIVAC version of POLAR. Of course, it is always possible to draw the plots using the graphics post-processor (Section 6.6).

In the S-CUBED UNIVAC version of POLAR, several plotting devices are available. They are the electrostatic plotter, the Calcomp and the Tektronix 4014. To automatically draw plots on the later device, one must run VEHICL from the Tektronix where the plots are desired. The general form of the PLOTDEST command is

PLOTDEST destination

where destination can be blank, NONE, CALC (for Calcomp), ELEC (for electrostataic) and TEKT (for Tektronix 4014). Leaving the destination blank or using NONE results in no plots being drawn at the conclusion of VEHICL (again the post-processor still can be used); this is the default condition.

OTHER VEHICL KEYWORDS

The following keywords are useful, or necessary, when running VEHICL. They are also described in Section 6.7.

BATCH

This keyword causes the input to be read in a batch mode. The main effect will be felt when erroneous input is discovered. If this occurs while in the batch mode, VEHICL will abort with an appropriate message. The default input mode for VEHICL is interactive (see description of INTERACT below). An example is

BATCH

which places the input routines in their batch input modes.

COMMENT

See description of REMARK below. The two keywords are equivalent.

DEFAULT

DEFAULT sets the VEHICL default values. In general, the defaults are for no diagnostic output, no plots, a grid spacing of 1 meter, and an interactive input mode. The keyword DEFAULTS is equivalent to DEFAULT. An example is

DEFAULT

VEHICL automatically calls the default routine before soliciting input. This keyword is most usedful when using the interactive mode and the previous input has not been satisfactory, or in error.

END

The keyword END is used to signify the end of input to VEHICL. This keyword should always be used at the end of a runstream, but if an EOF (end-of-file) is encountered instead, VEHICL will still function normally.

An example is

END

No more input will be read at this point and VEHICL will begin operation.

INTERACT

The INTERACT keyword is used to place the VEHICL input routines in an interactive mode. This means that any errors encountered in the input runstream will generate an appropriate error or warning but will not cause VEHICL to terminate its execution. This is the default mode of input for VEHICL. To abort on the discovery of bad input, use the BATCH command described above. An example of the use of the keyword is

INTERACT

which will place VEHICL in an interactive input mode.

REMARK

This keyword is used to insert comments in a runstream. When a REMARK is encountered, the remainder of the input card will be ignored and a new card will be read. Any number of REMARKs may be used. An example of the use of the REMARK keyword is

REMARK THIS IS A REMARK

All of the data on the card following the first REMARK will be ignored. The keyword COMMENT (mentioned earlier) can be substituted for REMARK and is completely equivalent, for example,

COMMENT THIS IS A REMARK TOO.

And again, everything on the card which follows COMMENT will be ignored.

TABLE 6.21/1
SUMMARY OF THE VEHICL KEYWORDS

BATCH	Places VEHICL input routines in a batch input mode. The default mode is interactive. (See INTERACT.)
COMMENT text	Causes text to be ignored by the input routines. Text may be any set of characters and numbers.
DEFAULT DEFAULTS	Resets options to default values. Called automatically at beginning of VEHICL.
DXMESH length	Defines grid spacing. Length is the grid spacing in meters. Default is 1 meter.
END	Makes the end of the VEHICL input. Should be included at the end of all runstreams.
INTERACT	Places VEHICL input routines in their interactive input modes. This is the default mode. (See also BATCH.)
MAKEPLOT option	Controls number of perspective plots. Valid options are DEFAULT (produces 6 default viewing directions) or an integer from 0 to 8. The default option is 0.
MAKEPLOTS option	Control material plot production. Valid options are YES and NO. The default is NO.
NXYZ ix iy iz	Defines object grid size, where ix, iy, and iz are integers defining the number of nodes in the x, y and z directions, respectively. This keyword must be included in all VEHICL runstreams.
PREFIX name	Defines the file name containing the object definition. If the file name is CUBEOBJ, name would be replaced by CUBE. (Keyword applies to UNIVAC only.)
PLOTDEST option	Controls where plots are drawn at end of a VEHICL run (UNIVAC only). Valid options are blanks, NONE, CALC, ELEC, and TEKT. The default is NONE.
PLOTDIR x y z	Describes viewing direction, from infinity, used to draw a perspective plot.
REMARK text	Causes text to be ignored by the input routines. Text may be any set of characters and numbers.

6.22 VEHICL DIAGNOSTIC KEYWORDS

There are several levels of diagnostic output available from VEHICL by keyword instructions. None of it is of interest to the casual user and is mainly a remnant of the code development process. But sometimes errors, code modifications or just idle curiosity will require some of VEHICL's diagnostic output. By default, all VEHICL output flags will be turned off or set to the lowest possible values.

The following is a description of the appropriate diagnostic flags and their settings.

DIAG General VEHICL Output

=0	No output.
=1	A few crucial tidbits from the construction of the A-2, KSURF, LCEL, and LTBL lists.
=2	More details concerning the construction of the various lists.
=3	Still more information.
=4	Provides a great amount of details concerning VEHICL's actions.

IDIAGS(1) DCVCEL Information

=0, 1, 2, 3	No output.
=4	Output from DCVCEL during the creation of the SREL list. Also, information from the various VCUBE routines.
=5	Additional DCVCEL data.

IDIAGS(2) CBUF Data Management

=0, 1	Nothing.
=2	Output from BUFSET.
=3	Output from PAGER.
=4	All of the information from PAGER and GRIDIO.

JDIAGS(1) SREL Information

=0, 1	No output.
=2	Print out LCEL and SREL lists. Also information from RECELL, SREL index to LCEL.
=3	Print out preliminary LCEL list.
=4	Detailed output from routines constructing LCEL and SREL lists.

JDIAGS(5) Double Point Data

=0, 1, 2	Nothing.
=3	Double point locations from SPDPNT.
=4	More double point information from SPDPNT.

JDIAGS(9) MTLGEN Information

<4	No information.
=4	Material properties and final PSGM matrix.

KDIAGS(1) MSIO information (UNIVAC only)

=1	MSIO routines echo calls to themselves.
=2	Perform tracebacks when called.

OBJPRT level	Output from initial object definition routines. Acceptable level keywords are NO, SOME or ALL.
HIDPRT level	Output from hidden line routine, HIDCEL. Valid level keywords are YES and NO. Note that YES will generate a vast amount of output.
IOGRID level	Grid information. Valid level keywords are YES and NO. Currently, this keyword is deactivated.

Some examples are

```
DIAG  3
IDIAGS(2) = 4
JDIAGS(9)  3
OBJPRT  SOME
HIDPRT  NO
```

This set of keywords would produce a high level of general information concerning the mechanics of the object definition (DIAG and OBJPRT), no MTLGEN or HIDCEL data (JDIAGS(9) and HIDPRT), and all of the available CBUF information (IDIAGS(2)).

6.23 AN EXAMPLE OF A VEHICL RUN

In Section 6.13, an object was defined (Figure 6.1/18) and a 3-D view of it was presented (Figure 6.1/19). Assuming the object definition kept in a file called XMPLOBJ (UNIVAC), or on file 20 (CYBER), and the necessary files have been declared large enough (6.20), Figure 6.2/1 shows the runstream (UNIVAC) used to draw Figure 6.1/19. The view used in the figure was the second one defined,

```
PLOTDIR  1.5  1.  .5
```

```
    1:@XQT VEHICL
    2:BATCH
    3:MAKEPLOT 2
    4:PLOTDIR 1.5 1. .8
    5:PLOTDIR 1.5 1. .5
    6:REMARK PLOTDIR 3. 1. 2.
    7:REMARK PLOTDIR 3. 5. 2.
    8:REMARK PLOTDIR 2. 1. -3.
    9:REMARK PLOTDIR 2. 3. -1.
   10:PLOTDEST NONE
   11:MATPLOTS NO
   12:DIAG 1
   13:JDIAGS(1) 2
   14:JDIAGS(9) 4
   15:REMARK IDIAGS(1) 2
   16:NXYZ 16 16 16
   17:DXMESH 1.
   18:PREFIX XMPL
   19:END
EOF:19
0:)
```

Figure 6.2/1. VEHICL runstream used to draw Figure 6.1/19.

In addition, several diagnostic flags were used which produced about 2000 lines of unnecessary output.

The grid was chosen to comfortably hold the object. If this object had been defined to be run later using NTERAK, the grid would have been defined with more care with respect to the final desired size. It is best to keep from defining a great deal of extra space in the z-direction (or in the direction which becomes the z-direction after reorientation) to prevent the needless IO of oversized object grid size tables (Section 5.3).

6.24 TROUBLESHOOTING VEHICL

Usually VEHICL will provide an explanatory error message before it dies. The most common fatal error which currently has no message or at least an opaque, user hostile message, is from DCVCEL in SUREAL. The message, "#####-FATAL-FROM DCVCEL - ELEMENT NUMBER ... DOES NOT CORRESPOND TO LIST ENTRY .....", results from the grid being defined too small or the object being too close to one side.

What has happened is that somewhere the object has touched the edge of the object grid. The cure is simply to try again with a larger object grid.

Poorly defined objects or ambiguous double points also create difficult problems. Some useful advice can be found in Section 6.11. If the definition process has proceeded far enough, it may be possible to produce material or perspective plots as a visual aide (Sections 6.20 and 6.21). The SHONTL module may also be used to draw the object or to print out some of the list output which has been saved in the MSIO files using CBUF storage functions (5.30).

6.30 ORIENT

In order that the object may be defined in an arbitrary direction or defined once and used in several orientations, it is necessary to be able to automatically reorient the satellite. The ORIENT module does this by rotating both the object and the object grid. This is necessary since NTERAK assumes the largest component of flow vector will be in the positive Z-direction, forcing a preferred direction on the problem. This module allows the same object to be studied in any flowing plasma without redefinition. Of course, if no rotation is necessary, OREINT does not need to be used.

ORIENT uses as input files the MSIO output files, 11 and 19, created by VEHICL. These files are also the output files, so they should be copied if the VEHICL output is to be preserved.

6.31 ORIENT KEYWORDS

In general, the keywords which were accepted by VEHICL (6.2) can be used in ORIENT since the two modules share many routines. Instead of redefining VEHICL keywords again, a reference to the section explaining the command will be given. The reader is also referred to the general keyword input section, 6.7, for more information. Table 6.31/1 contains a brief summary of ORIENT commands for convenience. The following is a description of the set of ORIENT keywords.

VMACH

VMACH is the plasma flow direction normalized to the ion acoustic speed. The entire problem will be rotated so that the largest component of VMACH will be in the positive z-direction. The rotated VMACH will be stored and will be the default value for the NTERAK module. But the Mach vector can be changed later to any value so long as the largest component is still in the positive direction. An example of the command is

```
VMACH  0.  0.  -8.
```

This would cause the object to be inverted so that the old negative z-direction becomes the positive z-direction. There is no default value for this command. If it is not defined ORIENT terminates with an error message. (This implies a default value of (0., 0., 1.), no rotation.)

For a description of the following ORIENT keywords see Section 6.21.

BATCH
COMMENT
END
INTERACT
REMARK

TABLE 6.31/1
A SUMMARY OF THE ORIENT KEYWORDS

BATCH	Places ORIENT input routines in a batch input mode. The default mode is *interactive*. (See INTERACT.)
COMMENT text	Causes text to be ignored by the input routines. Text may be any set of characters and numbers.
END	Marks the end of the ORIENT input. Should be included at the end of all runstreams.
INTERACT	Places ORIENT input routines in their interactive input modes. This is the default mode. (See also BATCH.)
REMARK text	Causes text to be ignored by the input routines. Text may be any set of characters and numbers.
VMACH x y z	Defines the desired orientation by the sign and location of largest absolute component. x y z is the three vector describing the Mach vector normalized to the ion acoustic speed. This keyword must be defined for ORIENT to execute.

Diagnostic output from ORIENT is obtainable by using the VEHICL diagnostic flags defined by Section 6.22.

6.32 RUNNING ORIENT

After the VEHICL output files (or a copy of them) have been assigned, the runstream shown in Figure 6.32/1 could be used to rotate the satellite defined by Figure 6.1/18 and pictured in Figure 6.1/19 so that neither of the two spheres point in the ram direction. After execution, the reoriented object can be viewed using the SHONTL module to see where in the grid the object ended up.

```
@XQT  ORIENT
BATCH
REMARK  ROTATE FIGURE 6.1/19 TO BE
REMARK  SIDEWAYS
.MA-   -1.  0.  0.
REMARK  THAT WAS EASY!
END
```

Figure 6.32/1 Sample ORIENT runstream.

Because the exact center of the grid is not necessarily on a node, the object may move a grid space in one (or more) directions. This can cause ORIENT to find an error during the SREL list creating. The error is due to the object touching the object grid boundary at some point. As previously mentioned (6.24), this error is cured by expanding the object grid. Unfortunately, only VEHICL can do this. So object must first be redefined in a larger grid by VEHICL, then rotated by VEHICL. It is a good practice to add an extra node on all sides of the grid if reasonable when an object is to be rotated.

An even better solution is to use an odd number of nodes along each axis. Then the center of the grid will fall on a node and the object should not touch the grid boundary in ORIENT if it did not in VEHICL.

6.45. SUMMARY OF NTERAK KEYWORDS

NTERAK is the program that calculates the plasma interaction and vehicle charging. Its keyword control language has both 'verbs' and 'noun'-options. A run-language sentence is constructed in the reverse-Polish sense, i.e., nouns first, verbs last. The NTERAK verbs are:

PWASON - invokes the Poisson solution
CURREN - invokes the sheath particle tracking
CHARGE - invokes the circuit model update of surface potentials
ENDRUN - run finished

The following is a list of the NTERAK keyword options. The first column is the keyword, the second column lists the possible responses. Literal options are listed with the default underlined. Numerical responses are indicated by the default value in parentheses followed by the variable type I, for integer; F, for floating point real number, E, for exponent type, i.e., 1.2E12, and L, for literal (no quotes).

OPTION	VALUE, FORMAT	EXPLANATION
BATCH INTERACT		Keyword only to select operation mode, turn prompts on or off, etc. non-critical.
DEFAULTS		Set or reset default option.
ISTART	NEW	New run. Allows for the calculation of new GI's or the use of old ones if the computation grids are identical. Space potentials will be zeroed, and a precharge step (automatic call to CHARGE) performed according to CHARGE options if requested.
	CONT	Continue a run, use old densities.
IGICAL		Ignored when ISTART ≠ NEW. YES - calculate new GI's (expensive). Default for ISTART = NEW. NO - Use 1.0's everywhere. OLDI - Use old DION's from previous run. Mach vector and computational grid should be identical. Default for ISTART = CONT. OLGI Use old GI's from previous run.
SAVSET		YES Equivalent of IGICAL = NO overrides IGICAL = YES with exception noted by GISAVE. NO Leave IGICAL = YES alone, SAVSET can be switched at location indicated by GISAVE.
GISAVE	X,L # , I Y,L # , I Z,L # , I	Two fields following the keyword. X 4 means that at all nodes in the plane at X = 4, the value of SAVSET will be temporarily toggled. Up to nine planes may be entered.

The combination IGICAL YES, SAVSET NO, and GISAVE Z -3, GISAVE Z -4, would cause the neutral ion density calculation (GI's) to be calculated everywhere except at Z = -3, and Z = -4. IGICAL YES, SAVSET YES, GISAVE X 4, GISAVE Y 5, GISAVE Z 12, would cause GI values to be calculated at the point 4 5 12 only.

OPTION	VALUE, FORMAT	EXPLANATION
EFLDCOR	YES NO	Use electric field corrections to the ion density calculations.
IGIOUT	NO YES	Output of appropriate densities.
PRECHG	PRE	Starting surface potentials from a precharge step.
	CONS	Constant surface potentials. Default for ISTART=NEW.
POTVAL	(V_{c1}), E,F	Initial insulator potential. Default is potential of conductor 1 (or V_{c1}).
CONDV	N pot	Initial conductor voltages where N is the conductor number (N=1 is ground conductor) and pot is the initial potential in volts.
INSULPOT	DIFF CONS	Defines POTVAL to be the initial DIFFerential voltage from insulator surfaces to underlying conductor or a CONStant insulator surface voltage.
FLOAT	n	Floats conductor n. If n (a conductor number) is omitted, all conductors are floated (the default condition).
FIXP	n V	Fixes potential of conductor n to V volts.
BIAS	n V	Bias potential of conductor n with respect to conductor 1 by V volts.
RIJ	i j r	Sets resistance between conductor i and conductor j to r ohms. For r < 1Ω, infinity (also default) will be used.
CIJ	i j c	Sets capacitance between conductor i and conductor j to c farads.
HELP		Output summary of option.
DXMESH	F,E	Grid spacing length in meters.
NXADNT	(0), I	Number of nodes added to the object grid in the +X direction to set up the computational space.

OPTION	VALUE, FORMAT	EXPLANATION
NXADNB	(0), I	Add nodes in -X direction.
NYADNT	(0), I	Add nodes in +Y direction.
NYADNB	(0), I	Add nodes in -Y direction.
NZADON	(0), I	Add nodes in -Z direction.
NZTAIL	(0), I	Add nodes in +Z direction (wake).
IOGRID	NO L YES	Output grid information.
REMARK COMMENT		Comment out. Runstream options.
NPHI	(36),I	Number of zenith angle divisions for GI calculation.
NTHETA	(180),I	Number of azimuthal angle division for GI calculation.
NADD	(2),I	Add extra vertices to object shadow in velocity space for GI calculations.
VMACH	VX VY VZ,F	Mach vector units of $\sqrt{kT/m}$. Default is set by VEHICL or ORIENT.
DENS	(1.0E11),E,F	Ambient density in $/m^3$.
TEMP	(0.2),E,F	Ambient temperature in eV.
DEN2	(0.0),E,F	For energetic Maxwellian in m^{-3}.
TEMP2	(0.0),E,F	For energetic Maxwellian in eV.
POWCO	(0),E, F	Power law coefficient $/m^2$.sec.str.eV.
PALPHA	(0),E, F	Power law exponent.
PCUTL	(1.0E2),E,F	Power law lower cutoff, in eV.
PCUTH	(1.0E9),E,F	Power law upper cutoff, in eV.
GAUCO	(0),E,F	Gaussian distribution coefficient $/m^2$.sec.str.eV.
ENAUT	(0),E,F	Gaussian peak in eV.
DELTA	(0),E, F	e-folding width of Gaussian.
RATIH	(1.0E4),E, F	Ion to hydrogen ratio.

OPTION	VALUE, FORMAT	EXPLANATION
AMUION	(16.0),E,F	Ion mass, in AMU
BMAG	(.4),E,F	Magnitude of B field, gauss.
BDIR	(-1 0 0)	Vector direction of B field, normalized by code.
BFIELD	OFF L ON	Turn magnetic field effects on and off.
FOR PWASON		
MAXITS	(2),I	Space charge iteration limit.
MAXITC	(10),I	Potential iteration limit.
MINITC	(2),I	Potential iteration lower limit.
RDRMIN	E, F	Potential convergence test. The default value is number of nodes * .0001.
POTCON	(6)	= log 10 (RDOTR(1)/RDOTR(ν)), potential convergence test.
IOCONG	PART L NO FULL	Output control of potential calculation.
ISPOUT	PART L LAST NO FULL	Output control of space charge calculation. To get final potentials from PWASON.
PDIE	(20*TEMP)E,F	Maximum allowable positive voltage in screening calculation.
SQALPH	(3),F	Space charge distribution.

OPTION	VALUE, FORMAT	EXPLANATION
FOR CURREN		
STHPOT	PSIM E, F	Particle sheath boundary potential. PSIM = $\ell n(SQALPH*(\lambda_D/DXMESH)^2)$
NTABLE	(10),F	The number of entries in the table used by SHEATH to calculate presheath weights.
IPCNT	(3),I	The maximum number of left and right particle pushing sequences allowed.
CURPOT	(-.45*TEMP), E,F	The potential at which the presheath weights are calculated.
FOR CHARGE		
MAXITT	(2) I	Number of timesteps.
DELTAT	(No default)E,F	Timestep, seconds, no default.
DVLIM	(1000.)E, F	Voltage change per step, in volts.
IMPEXP	IMPL / EXPL	Normal explicit-implicit sequence. Explicit only.
VLTFIX	(V_F),E,F	Estimated potential crossover for non-charging surfaces. (Used as a boundary on calculation.) V_F = (AMUION*$1.67\times10^{27}/9.1\times10^{31}$).
VFIXHI	(-10000),E,F	Estimate of upper (charging cases) crossover potential. Used as a boundary to try and limit voltage swings. Volts.
DVTEST	(5),E,F	Guess used in first CHARGE cycle for voltage change in previous timestep. Volts.
VWIGGL	(.001),E,F	Minimum change forced when a surface is near its equilibrium.
XDVFAC	(2),E,F	Multiplies DVLIM to be used in first trial of a CHARGE cycle to guess the potential of the crossover point.

OPTION	VALUE, FORMAT	EXPLANATION
FOR DIAGNOSTICS		
IDIAGS(#) JDIAGS(#)	#,I #,I	Diagnostic and output controls, see Section 6.45.10.
KDIAGS(#) IBIAGS(#)	#,I #,I	
OUTPUT	sec quant	Sets a set of DIAG flags appropriate to subsection (sec) and quantity (quant). Valid subsections are TIMER (run time information), PWASON, CHARGE, CURREN, and NEUDEN (the neutral ion density calculation). Valid quantities are HIGH, LOW, and OFF. Defaults are set to LOW for all subsections.
SELECT	options	Used to select specific Z-slices from large tables for output. Also can be used to turn output of a data set off (or on). The various options are described below.
SELECT	name list ENDLIST	Name is a buffered name (5.30) which is sliced. list is a set of integers separated by blanks which are object grid Z-slice values (e.g., to get IZ=2 and 3 of POT say SELECT POT 2 3 ENDLIST)
SELECT	name ALL/NONE	Turns on or off output of name (a buffered set of data). Default is on (ALL).
SELECT	LIST n list ENDLIST	Defines LIST n ($1 \leq n \leq 5$) as list (see above).
SELECT	name LIST n	Prints LIST n Z-slices when name data is printed.

6.45.10 NTERAK DIAGNOSTICS AND OUTPUT CONTROL

The diagnostic keywords recognized by NTERAK are IDIAGS(N) and JDIAGS(N). For example, a runstream might contain the card

IDIAGS(9) 1

All diagnostics have a default level of zero. Diagnostic flag levels are usually recognized in a > sense; i.e., larger flag levels include all the lower levels.

The diagnostic flags are as follows:

IDIAGS(1) used by PWASON
- = 2 CONGRD output
- = 3 COPROD output
- = 4 Additional COPROD info, all of the PCUBE routines, and DCVCEL output
- = 5 Additional DCVCEL output

IDIAGS(2) CBUF usage
- = 2 Information from DEBUF, BUFSWP, GETLAD, BUFSET (MORCOR calls, IAVECS settings)
- = 3 PAGER (TIMER calls on entry and exit)
- = 4 All info from PAGER and GRIDIO

IDIAGS(3) VERTIO check
- = 4 All VERTIO action

IDIAGS(4) TIMING
- = 1 in OPTIN
- = 2 in CHARGE, IONDEN, CONGRD
- = 3 in PAGER

IDIAGS(5) Self diagnostics

- =2 GRIDIO (calls CHKPUF to check the addressing system). IONDEN plus others possibly in future (calls CHKLOC to issue warnings if addresses within CBUF are too big to be passed within a word.)
- =3 CHKLOC prints core locates of the all locations passed to it.
- =4 CHKLOC generates non-fatal walkbacks for warning situations.
- =5 CHKLOC generates non-fatal walkbacks for all calls to it.

IDIAGS(6) Trajectory tracing (CURREN)

- = 1 Trajectory progress checking
- = 2 PRNTSL (particle reading and writing) action, particle movement at the slice level
- = 3 Complete output from PRNTSL, print out the various grids used by CURREN on exit
- = 4 Print individual particle movements
- = 5 Energy checking diagnostics

IDIAGS(7) QSELT

- = 4 QSELT information

IDIAGS(8) CHARGE

- = 0 Echo input
- = 1 Control flow info
- = 2 Voltage results from ICCG
- = 3 Input to ICCG, previous cycle data, material info
- = 4 Intermediate subroutine output
- = 5 Surface by surface calculations
- = 6 FLUXEL and FLUXBK output, CHKIMV surface checking

IDIAGS(9) IONCUR

= 1 Print SRFI list

= 2 Intermediate lists

= 3 Weight calculation results

= 4 Weight calculations

= 5 Particle lists from CURRENT

JDIAGS(1) SREL (VEHICL)

= 0,1 No output

= 2 Print out LCEL and SREL lists. Also information from RECELL, SREL index to LCEL.

= 3 Print out preliminary LCEL list

= 4 Detailed output from routines constructing LCEL and SREL lists.

JDIAGS(2) SHEATH calculation

= 3 Print presheath information

JDIAGS(3) PRECHG

= 2 Print ambient ion currents calculated by AMBCUR

= 3 ASRF from ALSURF

JDIAGS(4) IONDEN

= 1 Print progress notes

= 3 SHFTIT results (for filled or partially filled elements)

JDIAGS(5) Double point data

= 2 Potentials from POTSET

= 3 On input to major modules and double point locations from SPDPNT (vehicl)

= 4 More double point information from SPDPNT (vehicl)

JDIAGS(6) RHOI values

= 2 Print RHOIs after completion of CURREN (in CUEXIT)

JDIAGS(7) Particle pushing error control

= 1 Print info for particle in error

= 3

= 4 Kill run if 5 percent of particles are lost to errors

= 5 Die instantly when an error occurs

JDIAGS(8) Modified CHARGE output

= 1 Final surface voltage for the cycle, including INISET

= 2 Initial FTOT and FSM used (total current and total secondary current)

= 3* Components of current for each surface

* Note that the information produced by JDIAGS(8) = 2 is included in JDIAGS(8) = 3 so the flags are exclusive.

JDIAGS(9) MTLGEN (in VEHICL)

= 2 Print calculated AMAT

= 4 Print all appropriate information

JDIAGS(9) IONGEN (in NTERAK)

= 2 IONGEN prints DIONs

= 3 IONGEN prints LTYPs

KDIAGS(1) MSMODS

= 1 Routines echo calls to themselves

= 2 Perform tracebacks when called to find caller

KDIAGS(2) CHKSTH, CHKTRJ

= 3 Echo commands

KDIAGS(3) E field correction to geometric ion densities

= 2 Print tables which will be used

KDIAGS(4) DBLCHK (particle validation)

= 2 Potentials for rejected particles

= 4 Position velocity for all particles

IBIAGS(1) Magnetic field diagnostics from CURREN

= 0 None

= 1 Some

= 2 More

> 2 Lots

IBIAGS(2) Magnetic field information from CHARGE

= 0 None

= 1 Some

= 2 More

> 2 Lots

6.50 OPERATING SHONTL

SHONTL is POLAR's general plotting and information extraction program. Its primary sources for these functions are the two mass storage files, 11 and 19. For certain functions, SHONTL will actually duplicate NTERAK calculations.

SHONTL's primary plotting mode is to construct two-dimensional contour plots of data slices perpendicular to any of the three coordinate axes. Currently, the following variables may be viewed in this fashion: neutral ion densities ('GI's), composite ion densities ('DION's), stabilized charge densities ('QUSD'), and potentials ('POT'). In addition, SHONTL can execute the CURREN module of NTERAK and superimpose the results on any of the contour plots. In particular, there are two types of graphical output from SHONTL-CURREN. The sheath location can be indicated by a contour of X's using either the sheath potential that was used by NTERAK-CURREN, or optionally changed with the STHPOT keyword (Section 6.52); also, ion trajectories may be displayed with the following options: A total perspective of complete trajectories projected onto the plot's slice (keyword PERSPC), the portion of all trajectories that pass through the plot slice, trajectories emanating from a ring of sheath points lying in selected X-Y plane (X-Y only) (keyword RING) which can also use the PERSPC option, and in addition to all of the above the WEED option will throw out about 75 percent of the trajectories to reduce plot cluttering and computing costs. Finally, a single point trajectory option is also available (keyword SPOTS).

Other plotting features are limited to a spectrum plot of the energetic electron environment. Future modifications will allow SHONTL to reproduce the material plots and the hidden line object perspective views that are currently available only from VEHICL.

In addition to plotting, SHONTL can output to a printer (or whatever) the current values of virtually any variable stored on files 11 and 19, through the use of the PRINT NAME keyword where NAME would be the exact FORTRAN name of any variable known to MRBUF (a subroutine). A list of these can be found in Section 5.33. The format for this output is generally identical to that used by VEHICL, ORIENT, and NTERAK when these variables are output from these modules. A note of warning - variable aliasing is not always identical between modules. For example, potentials would be output by NTERAK by the 'card'

```
ISPOUT  FINAL
```

which would produce potential array output (modified by the SELECT keyword) at the conclusion of a spacecharge-potential calculation (the PWASON submodule of NTERAK). Whereas in SHONTL one would enter the cards

```
SELECT - options -
PRINT  POT
```

to get the same output.

Like the other modules, SHONTL keywords fall roughly into two categories that can be described as verbs and adverbs. For example, a user might enter the following adverb cards to set up plot parameters:

```
PLOTS ON
LEVELS 20
LEVELS  NOMARK
GRDPTS ON
SHEATH ON
```

followed by the verb card

```
POT  Y
```

which would trigger the actual production of a potential contour plot with a sheath indicated, no contour markings, approximately 20 contour levels, and all grid points indicated.

Finally, plots (but not output) are generated in a machine independent fashion on file 2., and a post-processor (usually PLOTREAD) is used to interface to the local graphics package. Depending upon installation and requirements, the user might want file 2 to be a permanent file.

6.51 STEP BY STEP INSTRUCTIONS FOR SHONTL

Step 1: Two data files are needed by SHONTL. They are file 11, and these files must be named 11 and 19. The best way to do this on the UNIVAC is to use the @USE command.

Step 2: Enter @XQT POLAR*SHOABS.SHONTL.

Step 3: Now you need to enter the information needed to retrieve the slice you wish to plot. The instructions for the input routine are in the form of keywords (6.52). Additionally, there are several commands to help use the keywords to get the desired plots (6.52).

The first step is to choose the plotting features you wish to use. These include marking the grid boundaries, showing the silhouette of the object, marking the grid points, choosing the axis for the problem and choosing a title for the plot. (There is a complete list of the keywords in the next section.)

Step 4: When all of the desired features have been chosen, a slice needs to be defined. At this point a plot is generated. There are several useful defaults which produce sets of plots (see 6.53). Now one can return to step 4 to produce additional graphs or continue onto step 5.

Step 5: To leave SHONTL, type EXIT if you want hard copies of the plots to be generated. Type ESC (for escape) if no hard copies are desired. The EXIT command will call PLOTRD automatically.

Step 6: Show all of your friends your nifty plots.

6.52 SHONTL KEYWORDS

The following is the current list of SHONTL keywords. All of the input which cannot be recognized is simply printed on the screen and then ignored.

The form of a keyword definition is

KEYWORD[, equivalent forms](*default setting*/other settings)
definition

The terms "ON" and "OFF" describe the setting of the option. So if something is set to "OFF", it will not appear on the plot. For example,

PLOTS OFF

will cause those plots generated while the flag is off, to not be printed. Since default value of PLOTS is off, to prevent undesired plots, the definition of PLOTS looks like

PLOTS (*OFF*,ON) causes the .....

Here is the present keyword list:

ALL (ON/OFF) turns all of the on/off switches to ON or OFF. "ALL ON" would turn on all of the keywords which can be set to (ON/OFF).

AXDRAW (*ON*/OFF) controls the printing of the plot axis, scale values, and axis labels.

A2SURF (*ON*/OFF) controls the printing of the object's silhouette.

COLOR (ON/*OFF*) turns on color filled features. This may depend on the use of a Jupiter.

COMMENT causes the input routine to ignore the rest of the line. Note that only the first six characters on the line are actually read.

CONLIN (*ON*/OFF) controls the printing of the contour lines on the plot.

DEBYE causes the scale on the plot to be in Debye lengths. The origin (0,0) is the object grid origin (see 'GRID'). The normal default is to grid units.

DENSIT,DEN,D. This is a request for a plot of a slice of density data. The allowable forms of requests for density slices are

```
DENSIT X N
DENSIT X MIDDLE
DENSIT X
DENSIT
```

where either of the abbreviations may be substituted for DENSIT. Also Y or Z can be used for X, MID can be used for middle, and N is a positive integer on the axis (X, Y, or Z) that is in the grid space. The axis is the axis perpendicular to the plane of the slice. If form 2 or 3 are used, the middle of the object grid will be plotted. If the last form is used, the middle X and Y slices will be plotted.

ERASE causes any plots which have been created to be erased and not produced by PLOTRD.

ESC causes the SHONTL module to be exited immediately without any additional output. (The keyword ESCAPE is equivalent.)

EXIT signals the end of the input. If no plots have been requested at this point and 'PLOTS' is 'ON', then four plots will automatically be produced. They will be plots of the potential and density at the middle of the X and Y axes.

FILLIN (*OFF*/ON) causes the interior of the object to be filled. The routine looks for zeros surrounded by relatively big numbers. This changes the appearance of the contour lines in a way which may be desirable. (Obsolete.)

FRAME (*PLASma*/CRAFt) - (**/X Y Z). This keyword chooses the reference frame for viewing potentials. The default is the plasma frame used by NTERAK, but a spacecraft frame may be chosen where the $\vec{v}x\vec{B}$ electric field will appear at 'infinity'. In the spacecraft frame the definition of plasma ground will appear arbitrary (it is not really) but it may be shifted from the point chosen by NTERAK.

GETSLC (*ON*/OFF) causes the plot generation routine to read a slice from the data file.

GRDPTS (ON/*OFF*) causes crosses to be printed at each of the nodes.

GRID causes the scale of the plot axes to be in object grid units. The reference point in this coordinate system is the lower left corner of the object definition grid which is defined to be (1,1). This is the default scaling.

HEADNG (*ON*/OFF) causes the heading to be printed at the top of the plot.

IDIAGS(N) is the keyword input section which allows setting of any of the IDIAGS flags. For information concerning the IDIAGS, see Section 6.70.

INPSPT (*0*). The keyword, INPSPT (INPUT SPOT), is used to trace the trajectory of a number of particles (SPOTS) which the format for keyword is "INPSPT N", where N is the number of points to be entered. The maximum number of points is 20. After the INPSPT keyword, the N particles are defined by giving their initial position (X,Y,Z) and velocity (VX,VY,VZ). For example, if we wanted to define two particles, one with the position (4.,5.,6.) in grid units and velocity (-.1,-.1,-4.) normalized to the ion acoustic speed (see documentation, POLAR Manual, Section 5.62.12), and a second particle with the position (-1.,3.,-2.5) and a velocity of (-1.,-.3,.01) we would enter:

```
INPSPT 2
4.  5.  6.  -.1  -.2  -4.
-.1  -.3  -2.5  -1.  -.3  .01
```

JDIAGS(N). See entry for IDIAGS(N).

KDIAGS(N). See entry for IDIAGS(N).

LEVELS (*10*-AUTO) set the number of contour lines to be drawn and their selection mode. The keyword LEVELS is also an entry word to the following contour options:

LEVELS AUTO — linearly spaced contours, rounded to nice numbers, for potentials, the kT and 0.1*kT contour will be added and the kT level marked.

LEVELS NOMARK turns off all contour marking.

LEVELS MARK VALUE MARKER — will mark the contour level at VALUE with the first character of MARKER. Marking with a number can be accomplished by #LLL where # is a number and L is any letter. Only # will be used, but the LLL is necessary for # to be recognized as a literal. This command may be repeated to mark up to nine levels.

LEVELS ADD VALUE, add a contour level
LEVELS SUB VALUE, remove a contour level
LEVELS SELECT VALUE, turn off auto mode and use input levels

LINEAR means the contour lines will be of the unmanipulated data in the slice. This is the default mode.

LOG causes the contours to be of the natural log of the slice data. The normal default is 'LINEAR'.

LOG10 causes the contours to be of the base 10 log of the slice data. The normal default is 'LINEAR'.

METERS causes the units of the scale of the plot axis to be in meters. The origin (0,0) is the object grid origin (see 'GRID'). The units default to grid units.

MIDDLE,MID. These keywords must follow X, Y, or Z to be meaningful. In this context, they cause the plot to be of the middle of the object grid. (For a more exact description of the middle of the grid, see 'DENSITY'.)

NONE. This keyword can be substituted for POTENT or DENSIT. It will cause plots to be made without potentials or ion densities. It can be used to look at the output from VEHICL. When plots of sheaths and trajectories are made, it is also helpful. The syntax for NONE is exactly the same as for POTENT or DENSIT.

NUMBER (ON/*OFF*). NUMBER is used when tracing particles. If it is used in conjunction with IDIAG(6)=5, the particle pusher will number the particles for diagnostic output. This is very useful for debugging the CURREN segment. (See also the notes on IDIAGS.)

OBJCNR (ON/*OFF*) causes four small boxes to be drawn in the corners of the object grid.

OBJGRD (ON/*OFF*) causes the object grid to be plotted.

OUTREL (*ON*/OFF) causes the outer stepped grid of real nodes to be plotted.

OUTVIR (*ON*/OFF) causes the outer stepped grid of virtual nodes to be plotted.

PERSPC (ON/*OFF*). When PERSPC is on, all of the sheath points and trajectory paths will be plotted. If it is off, then only those points or paths which are in or cross the slice being drawn will be shown.

PLOTS (ON/*OFF*) causes the plots to be produced. If it is off, the slices will be read and processed for plotting, but will not be plotted.

PLTGRD causes the scale of the axis to be in plotting grid units. These are the same size as object grid units but the lower left corner is defined to (0,0). In object grid units, the reference point on the grid is the lower left corner of the object definition grid which is defined to be (1,1). The default is object grid units.

POTENT,POT,P. This is the same as 'DENSIT' except potential slices instead of density slices will be plotted. Please see 'DENSIT' for more information.

PRINT name causes name to be read from the appropriate file. Valid name replacements are any of the buffered data lists mentioned in Section 5.33. For larger data sets, the use of SELECT is recommended (see below).

PRTSLC (*OFF*/ON) causes a grid of integers to be printed. This is helpful to examine a slice for plotting before a plot is actually produced. Be sure to turn 'PLOTS OFF' if plots are not desired.

RESET. This keyword resets all of the plotting flags to their initial values.

RING (ON/*OFF*). Sometimes on large problems, it is desirable to trace fewer particles. One way to do this is just follow a ring of particles. Presently the only rings in the XY plane can be made. N is the Z slice of the ring. If a ring in the Z=22 slice was desired, the keyword would be "RING 22". If RING is followed by ON or nothing at all, the middle Z slice will be used.

SELECT options used to choose particular Z-slices for printing when using PRINT or a DIAG (or IDIAGS, etc.) flag. Many options are available and the user is referred to Section 6.44, 6.45 or 6.70.

SHEATH (ON/*OFF*) causes the sheath points to be printed on the plot.

SPOTS (ON/*OFF*). To push a few test particles to see which trajectories they will take, use the SPOTS flag. To enter these particles, see INPSPT. It is recommended that PERSPC is ON when SPOTS is used.

STHPOT N. The SHEATH potential for plotting the sheath can be changed by resetting STHPOT. The initial value is the same as the value used by CURREN during NTERAK. N is a real number.

TITLE. Used to define the title to be put on the plot. The default is "POLAR PLOTS". The desired plot title should start in the ninth column of the card

```
1234567890123456
TITLE  BIG BIRD
```

In the above example, the heading appearing on the plot would be "BIG BIRD". Currently there is a 16 character maximum. (Presently unimplemented.)

TRAJ (ON/*OFF*). This flag turns on plotting of particle trajectories.

WEED (*ON*/OFF). This flag reduces the number of particles usually found by CURREN by about a factor of 4. This not only speeds up the plotting process but also makes the plots a little clearer.

WHAT causes all of the print options and their current settings to be printed at the terminal.

X,Y,Z the axis perpendicular to the slice to be plotted. If both potential and density plots are desirable, something like

```
X 8
```

can be typed. This plots the X = 8 slice of the density and the potential data. Instead of 8, any integer containing part of grid, middle or mid, or nothing at all can be used. Nothing at all will default to a middle slice. Valid forms are

```
X N
Y MIDDLE
Z MID
X
```

where X, Y, and Z are interchangeable and N is any integer as described above. Please see 'DENSIT' for more information.

6.53 SPECTRUM KEYWORDS AND OPERATING INSTRUCTIONS

The spectrum plots are generated by SHONTL using the plasma characteristics used by a previous NTERAK run or by defining the environment using the same keywords used in NTERAK.

SHONTL keywords:

(Note that the range of values plotted on the vertical axis is currently not controllable by the user and is limited to six orders of magnitude down from the maximum flux while in the LGLINY LOG mode.)

Default options are marked by surrounding *'s. These *'s are not part of the keyword.

6.54 SHONTL DEFAULTS

SHONTL has lots of defaults. All of the plot features have defaults (6.52). There are also slice default values for which slices are printed. The current slice defaults are also in Section 6.52 under the 'DENSIT' definition.

The default values will vary as the plotting requirements change. The best place to look for default changes will probably be the code itself. The WHAT command gives the feature defaults.

KEYWORDS

HIGHEN	n	n is the value of the energy cutoff on the high end.
INKEV	NO *YES*	y axis always in eV, x axis is variable.
LGLINY	log *lin*	a log 10 or linear plot of data on the vertical axis.
LGLINX	log *lin*	a log 10 or linear plot of data on the horizontal axis.
NPTS	n	number of points to be calculated and connected ($0 < n < 200$).
RLOWEN	n	n is the value of the low energy cutoff on the horizontal axis.
SPECTRUM		draw a spectrum plot using the previously entered information.
WRITE		write the plasma environment generated so far during the current run onto files 11 and 19.

END

12-86

DTIC